N00014-89-C-0219

NOVEL NUMERICAL ALGORITHMS FOR SENSING, DISCRIMINATION, AND CONTROL

FINAL TECHNICAL REPORT
September 15, 1989 to March 15, 1990
Contract #N00014-89-C-0219

SPACE TECH CORPORATION
125 Crestridge Drive
Fort Collins, CO 80525
303 223-8166

Prepared for
Office of Naval Research
Department of the Navy
800 N. Quincy Street
Arlington, VA 22217-5000

This work was sponsored in part by the Office of Naval Research, Contract #N00014-89-C-0219. The views, opinions, and findings contained in this report are those of the author(s) and should not be construed as an official Department of the Navy position, policy or decision, unless so designated by other documentation.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER N00014-89-C-0219	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Novel Numerical Algorithms for Sensing, Discrimination, and Control		5. TYPE OF REPORT & PERIOD COVERED Final Technical Report 15 SEP 89 to 15 MARCH 90
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Dr. Michael Andrews		8. CONTRACT OR GRANT NUMBER(s) N00014-89-C-0219
9. PERFORMING ORGANIZATION NAME AND ADDRESS Space Tech Corporation 125 Crestridge Drive Fort Collins, CO 80525		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Office of Naval Research 800 N. Quincy Street Arlington, VA 22217-5000		12. REPORT DATE March 9, 1990
		13. NUMBER OF PAGES 50
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Wigner-Ville, tomography, computers, vector, photonic, phase retrieval

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes research and development of efficient algorithms and architectures for certain SDI processing tasks. Spin rate, phase retrieval, and superresolution were analyzed. Efficient mappings of pertinent convolution, Hilbert and Wigner-Ville transforms, and an SVD onto a novel parallel/pipelined architecture were established. An ultrafast E/O interface method to high speed electronic circuits is described. The proposed solution is a fast SIMD/MIMD machine capable of teraop speeds, combining multiple 4-PE horizontally microprogrammable nodes onto a mesh-connected organization (fiber-optics).

N00014-89-C-0219

NOVEL NUMERICAL ALGORITHMS FOR SENSING, DISCRIMINATION, AND CONTROL

FINAL TECHNICAL REPORT
September 15, 1989 to March 15, 1990
Contract #N00014-89-C-0219

SPACE TECH CORPORATION
125 Crestridge Drive
Fort Collins, CO 80525
303 223-8166

Prepared for
Office of Naval Research
Department of the Navy
800 N. Quincy Street
Arlington, VA 22217-5000

[illegible]
Dr. Michael Andrews, Principal Investigator

Table of Contents

EXECUTIVE SUMMARY

This report describes research and development of efficient algorithms and architectures for certain SDI processing tasks. Spin rate, phase retrieval, and superresolution were analyzed. Efficient mappings of pertinent convolution, Hilbert and Wigner-Ville transforms, and an SVD onto a novel parallel/pipelined architecture were established. An ultrafast E/O interface method to high speed electronic circuits is described. The proposed solution is a fast SIMD/MIMD machine capable of teraop speeds, combining multiple 4-PE horizontally microprogrammable nodes onto a mesh-connected organization (fiber-optics). Space Tech personnel involved with this effort are Michael Andrews and Joseph Webster. This study was sponsored by SDIO/IST under the direction of scientific monitor, Dr. Keith Bromley, ONR, NOSC.

1.0 Algorithm-Specific Architectures

1.1 Phase I Technical Results

The objectives of Phase I were to identify and analyze fast signal processing architectures for three important SDI tasks, namely, target spin rate detection/estimation, phase retrieval, super-resolution, determine superior device technologies, and determine the optimal algorithm mappings onto the fast architectures. These objectives have been met.

A number of critical architectures were examined for enhancing the concurrent computing of any inherent parallelism of the three major algorithms. These included bit-slice, microprogrammable, data-flow, vector, array, systolic, and very wide instruction machines. Anticipating technology growth by 1992 suggests that the TENSOR architecture as proposed in a Phase II effort will capture these opportunities if, among other techniques, an innovative crossbar which ties the arithmetic engines to the local storage and control units is developed.

The majority of the Phase I effort resulted in discovering a very wide instruction machine that is massively pipelined/parallel in operation, determining the HYPERBAR data paths for each algorithm, and designing efficiently fast alternatives. That innovative system architecture is called a STARBURST, capable of expanding to 64 TENSOR nodes via HYPERBAR in true parallel operation at Teraop speeds. Other architectures such as systolic, although maintaining significant sustained throughputs, could not generate peak throughputs often necessary in demanding signal processing applications such as SDI.

1.1.1 Phase I HYPERBAR Innovation

A major innovative result of Phase I was the ultrafast, synchronous HYPERBAR that enables the massively parallel execution of Hilbert, Wigner-Ville and arithmetic Fourier transforms in minimal physical space. Another innovative result was a PGA ASIC architecture for concurrent multi-dimensional addressing to accelerate matrix processing. A third major result of Phase I is the identification of an innovative single node desktop scientific workstation that supports the three major algorithms with sustained gigaflop throughputs for under $25k by 1992. This single TENSOR processor (TPH) node architecture of STARBURST is capable of manipulating 128 by 128 element matrices with 64 bit floating point arithmetic with CMOS devices (a relatively low risk technology). A single TENSOR computing node is superior to the class of SUN, Solbourne, and IBM 6000 RISC machines that cost over $100k (1990$) with speeds barely approaching 200 mflops peak. The possibility for this dramatic price reduction and performance increase is found in increased utilization of the low NRE ASIC and custom chip design, full use of firmware in the micro-grain architecture, and breakthroughs in the coupling logic elements.

1.1.2 Phase I Centerplane Innovation

The engineering analysis had to first identify a best architecture, parallelize the algorithms, optimize the dataflow paths, and minimize critical delays. Yet, this alone did not guarantee the low cost TENSOR workstation goal of 1992. Conservative prediction of technology device integration and speeds further required complex analysis of the packaging and cooling needs to minimize the board-to-board wire lengths. As a result, an innovative centerplane (in contrast to backplane) board level packaging technology was discovered. This new fiber optic design allows us to increase edge connections to support 64 bit parallel data buses directly. Furthermore, the current architecture designed using 1990 technology and requiring four boards can be easily reduced to one board using the predicted 1992 devices (SRAMs, DRAMs, FIFOs, and Multiplier/Accumulator pairs).

1.1.3 Phase I Address Generator Innovation

It is now clear that this new architecture with a strategically allocated STARBURST interconnect scheme and novel packaging densities can aggressively compute the closely-spaced object resolution and scan-to-scan target identification well beyond the speeds of the fastest neural net machines. The critical breakthrough was found in the ancillary circuits such as the address generator, register files, and local memories. In fact, this machine could very well be called a true **TENSOR processor** because multiple addresses (row, column, and elevation) are generated concurrently in hardware instead of firmware or software unlike any machine today. Furthermore, if the discrete address generator logic is placed in custom logic with BiCMOS or ECL, an additional twofold increase in speed becomes possible. (However, the economic cost rises dramatically.)

1.2 SDI Problem

The taxonomy of computing architectures is diverse and numerous. Selecting an "optimal" architecture for specific algorithms reduces the set. In this research, the investigators were seeking good mappings of a small set of algorithms onto a good universe of architectures. The goodness-of-fit test for each architecture was the overall speed of execution, throughput, and efficiency of VLSI. The goodness of each algorithm was a measure of its robustness to ill-conditioned data sets which is partially reflected in the wide dynamic range of the signal space. Extending the machine precision was only a partial solution. In all of the algorithm mappings, a solution was sought which handles 2-D and 3-D data.

For the specific set of algorithms necessary for sensing, discrimination, and control, better architectures were found. These are described along with the algorithmic mappings which "parallelized" any inherent concurrency of the algorithms. Certain ASIC oriented architectures were found that enhanced the throughput when the algorithms were partly recursive and sequential in nature. The architectural choice must be sensitive to the inherent parallel nature of the algorithms (or flowgraph), any modularity of the input data streams, data dependency characteristics of the primitive computational steps, and throughput. Some signal processing tasks may require very short latency. Hence, certain architectures will remain sub-optimal. In the sensing, discrimination, and control tasks of this study, latency was not nearly as important as throughput and was not considered as a significant design parameter.

Some general choices initially were made on the basis of the class of algorithms. The algorithm class is characterized by streaming dataflow, 2-D and 3-D data configurations as typically found in signal and image processing. Image processing spans several orders of throughput specifications. At the front end of the imaging data stream, individual pixel level processing for filtering, contrasting, contouring requires 100 MHz clock rates for 32-bit data arriving at 10-40 MHz frame rates. Imaging processing at the back end looks for patterns or "global" parameters and slower clock rates of 1 MHz suffice. This phenomena is typical of the signal compression path from pixel to data.

Any "optimal" architecture and algorithm map must address minimal features of the technology. In this study, the investigators assumed that CMOS 1.5 micron was readily available in 1990 with VLSI integration levels at 200,000 gates. By 1992, the start of a Phase II effort, it is anticipated that VLSI integration levels of 1,000,000 gates with .5 micron line widths are conservatively feasible. Commercial chip yields are assumed in both cases. For ASIC devices, only double level metallization was required. Even though this study sought optimal algorithms onto an optimal system architecture, forecasting the technology at the gate level was necessary. From this design perspective, architects can assess the speed of their maps ultimately by the speed of the clocking rate of the individual devices and bus transfer rates.

2.0 Algorithm Analysis

The specific sensing, discrimination, and control SDI tasks are spin rate estimation, phase retrieval, and super-resolution. The algorithms for the spin rate estimation are tomography, Wigner-Ville transform, Radon transform, and related algebraic reconstructions. The dominant algorithms for phase retrieval are the fast Fourier transform, Arithmetic Fourier Transform and the Chirp-Z transform. The main algorithm for superresolution is a regularization of certain data matrices. The signal processing bottleneck, however, specifically is a matrix inversion which is solvable by several methods. In this study, an LU decomposition with and without row/column interchange has been used. Others are as equally robust, but it is doubtful that they are superior to this method in TENSOR.

2.1 Spin Rate

Spin Rate Identification can be solved using positron emission tomography. Whitehouse has provided adequate assessment with algorithms similar to filtered backprojections and Radon transforms. Further work disclosed the utility of the Wigner-Ville (XWV) transform. The TENSOR architec-ture proposed herein is finetuned to rapidly compute the Wigner-Ville with minimum latency to successive time sequenced images. A cross WV with appropriate windows in TENSOR can identify spin rate for these nonstationary signals of bandlimited duration in real-time. **The fastest method is to configure parallel processors to compute several inner product sequences (one for the Hilbert transform and another for the WV).**

2.1.1 Tomography

Tomographic imaging can be done with several algorithms, including frequency domain, interpolation, backprojection, filtered backpropagation, and modified backpropagation. In addition the images can be reconstructed by several means such algebraic reconstruction, simultaneous iterative reconstruction, and simultaneous algebraic reconstruction. We already know that holography is not an exact inversion method because it often distorts the image [WuTo87]. Diffraction tomography is appearing to be more promising. The basic principle as shown by [Wolf69] assumes a scattered field described with Born approximations is used to reconstruct the distribution of the refractive index for a weakly scattering object.

In searching for simplifying architectures, a paper by Bernfeld and a follow-on by Snyder suggest that certain near realistic assumptions about scattering functions of practical targets may provide easier computational tasks. In fact, the computations amount to no more than deconvolutions. However, Bernfeld's analogy of delay-doppler to positron-emission tomography [Bern84] breaks down because the ambiguity function of the transmitted pulses must be highly concentrated along lines in the delay-doppler coordinates. In addition, the ambiguity function must have constant amplitude along those lines. Snyder, et al [Snyd86] ameliorate this problem by relaxing a restriction. The simplifying assumption is that the additive noise is negligible and the resulting observed data can be replaced by their average value. As a result, the imaging problem is a deconvolution. Briefly, given the ambiguity function and data for several angles, solve a double integral (deconvolution).

The computational model is structured as follows using Van Trees [Vant71]. If E_t is the square root of the transmitted power, the transmitted pulse has complex amplitude

$$E_t f(t) \tag{1}$$

The reflected pulse becomes

$$s(t) = \int E_t f(t-\gamma) b(t-\gamma/2, \gamma) d\gamma \tag{2}$$

where now, $b(t-\gamma)$ is a zero-mean, complex-valued Gaussian process modeling a diffuse reflection interaction at the target. The target's scattering function is then represented by the power spectrum of $b(t-\gamma)$ at a range, γ, which we denote as

$$\Omega(t,\gamma) \tag{3}$$

Now assume that $b(t,\gamma)$ models a wide-sense stationary, uncorrelated scattering process and the received signal $r(t) = s(t) + w(t)$ in which the additive noise, $w(t)$ is a complex-valued, white Gaussian process that is independent of b and has spectral intensity, N_0.

Snyder shows that, if $r(t)$ is processed by a bandpass matched-filter square-law envelope detector, the receiver output averaged value is $\sigma(f,\gamma) + N_0$, such that

$$\sigma(t,f) = E_t \iint \Omega(t,\gamma)\ a(\gamma-\gamma', f-f') d\gamma' df' \tag{4}$$

where $a(t,f)$ is the ambiguity function of the transmitted signal which is the squared magnitude of the complex delay-doppler correlation function. The computational process becomes one of estimating the target's scattering function from the received data $r_\theta(t)$ from a series of target illumination angles, θ, and solve Equation (4) for the scattering function, Ω. The major computational task is a deconvolution step (vector inner products again).

2.1.2 Wigner-Ville Transform

Absorption and dispersion effects upon elastic waves have already been studied in [Boua79] and shown that the possibilities of the Wigner-Ville Distribution (WVD). To improve resolution, we need to solve the problem of "a direct inversion using the wave equation". [Boas86] shows the efficiencies in increased resolution to be gained by the WVD, modified WVD [Boas87], and Cross WVD (XWVD) [Boas89]. A modified XWVD is especially useful because it minimizes the number of cross terms or ghosts that corrupt interpretation of the useful events. Without some additional computations the WVD alone does not provide safe interpretation because the cross terms fall between real events. If the signal strength is high, then these ghosts may even hide or cover useful returns. We do know that the cross terms can be distinguished from the genuine signals by their oscillating nature and that the cross terms are expected to greatly outnumber the genuine signals. A modified XWVD is especially useful because it minimizes the number of cross terms or ghosts that corrupt spin rate identification.

Autoregressive moving averages then could be used but the high order of these models increases computational burden. Only with signals of relatively low numbers of components (<6) was the method found beneficial. Boashah also proposed a modified short term Fourier transform to decrease computational complexity. But acceptable levels of artificial frequency dispersion were not achieved and should no longer be considered.

The only practical application of the WVD was in the implementation of the XWVD. The advantages include a relatively high resolution capability in the time-frequency distribution independent of the emitted signal. This independent nature is critical to analyses where the source signal may drift thus rendering complicated signal conditioning. And since only the real part of the complex signal is needed for interpretation, analysis is straightforward. Where active FM chirp signals are used, excellent event separability is accomplished in reflections. Hence, a **modified XWVD that ensures amelioration of the low frequency distortion is critical.**

The discrete time WD (DTWD) is defined as:

$$W(n,\theta) = 2 \sum_{k=-\infty}^{\infty} z(n+k)\overline{z}(n-k)\exp(-j2\theta k) \qquad (5)$$

where overbar indicates complex conjugation ($z\overline{z}$ must be analytic). In the sequel we will take

$$z(n) = s(n) + j\hat{s}(n)$$

where $\hat{s}(n)$ is the Hilbert transform of $s(n)$. Because only a finite amount of data can be processed, windowing has to be employed. The DTWD of the windowed data is:

$$W(n,\theta) = 2 \sum_{k=-M}^{M} z(n+k)\overline{z}(n-k)w(k)\exp(-j\theta k) \tag{6}$$

For simplicity without loss of generality, choose the rectangular window $w(k)=1$, $-M \le k \le M$. A different choice would result in only slight modification to any architecture mapping follows. The WD $W(n,\theta)$ can now be evaluated as a finite number of points $\theta = (\pi/N)m, m\in[0,N-1]$ that cover its basic period $\theta\in[0,\pi]$.

$$W(n,m) = 2 \sum_{k=-M}^{M} z(n+k)\overline{z}(n-k)\exp(-j2\pi mk/N) \tag{7}$$

$$= 2 \sum_{k=-M}^{M} r_n(k)\exp(-j2\pi mk/N)$$

where $r_n(k)=x_n+jy_n(k)=z(n+k)\overline{z}(n-k)$, $-M \le k \le M$. Usually, $N = 2M$

The auto-product $r_n(k)$ has a couple of symmetry properties that can be exploited in order to reduce the computation by a factor of 4. First,

$$r_n(k) = \overline{r}_n(-k), \quad -M \le k \le M \tag{8}$$

Only the real part of the discrete Fourier transform (DFT) of such a sequence will be non-zero. On the other hand, the DFT of the combination of two successive sequences

$$R_n(k)=r_n(k)+jr_{n+1}(k) = x_n(k)-y_{n+1}(k)+j[x_{n+1}(k)+y_n(k)] \tag{9}$$

will have:

$$\text{Re}\{\text{DFT}[R_n(k)]\} = \text{DFT}[r_n(k)]=W(n,m) \tag{10}$$

$$\text{Im}\{\text{DFT}[R_n(k)]\} = \text{DFT}[r_{n+1}(k)]=W(n+1,m) \tag{11}$$

Therefore, a single DFT computation on $R_n(k)$ will produce two slices of the DTWD:$W(n,m)$ and $W(n+1,m)$. This reduces the number of DFT computations by 2.

Another symmetry property:

$$R_n(-k)=\bar{r}_n(k)+j\bar{r}_{n+1}(k) = x_n(k)+y_{n+1}(k)+j[x_{n+1}(k)-y_n(k)] \quad (12)$$

implies that $x_n(k), y_n(k), y_{n+1}(k), x_{n+1}(k)$ should be evaluated only for non-negative k, $0 \le k \le M$. Therefore, the number of multiplications required to compute $R_n(k)$ can be halved by exploiting this property:

$$W(n,m) + jW(n+1,m) = 2 \sum_{k=-M}^{M} R_n(k)\exp(-j2\pi mk/N)$$

$$= 2R_n(0) + 2 \sum_{k=-M}^{M} R_n(k)\exp(-j2\pi mk/N)+R_n(-k)\exp(j2\pi mk/N)$$

$$= \{2x_n(0) + 4 \sum_{k=-M}^{M} [x_n(k)\cos(2\pi mk/N)+y_n(k)\sin(2\pi mk/N)]\}$$

$$+ j\{2x_{n+1}(0) + 4 \sum_{k=-M}^{M} [x_{n+1}(k)\cos(2\pi mk/N)+y_{n+1}(k)\sin(2\pi mk/N)]\} \quad (13)$$

As a result, we obtain the following computational sequence:

Given $s(i)$, $n - M \le i \le n + M + 1$, $p = 2M$

Repeat:

$$\hat{s}(n) = \sum_{k=0}^{M} h(k)[s(n+(p-1)-2k)-s(n-(p-1)+2k)]: \quad (14)$$

$$\hat{s}(n+1) = \sum_{k=0}^{M} h(k)[s(n+p-2k)-s(n-(p-2)+2k)]: \quad (15)$$

For m = 0,N-1

$$W(n,m)=1/2[s^2(n)+\hat{s}^2(n)], \quad (16)$$

$$W(n+1,m)=1/2[s^2(n+1)+\hat{s}^2(n+1)], \quad (17)$$

For k = 1,M

$$x_n(k)=s(n+k)s(n-k)+\hat{s}(n+k)\hat{s}(n-k), \quad (18)$$

$$y_n(k)=\hat{s}(n+k)s(n-k)-s(n+k)\hat{s}(n-k), \quad (19)$$

```
    $x_{n+1}(k)=s(n+1+k)s(n+1-k)+\hat{s}(n+1+k)\hat{s}(n+1-k),$        (20)

    $y_{n+1}(k)=\hat{s}(n+1+k)s(n+1-k)-\hat{s}(n+1+k)s(n+1-k);$        (21)

For $m = 0,N-1$

    $W(n,m)=W(n,m)+x_n(k)\cos(2\pi mk/N)+y_n(k)\sin(2\pi mk/N),$

    $W(n+1,m) =$                                                        (22)

    $W(n+1,m)+x_{n+1}(k)\cos(2\pi mk/N)+y_{n+1}(k)\sin(2\pi mk/N);$     (23)

$n = n+2;$

until DONE.
```

Because the DTWD is so vital to most of the SDI tasks, an ana'ysis of its computational speed is made on the TENSOR architecture (detailed later on). However, the basic architectural features are on MIMD or SIMD machines, mesh-connected with 4 FPUs, 4-port local memory, large register file, and a full crossbar to all of the previous resources for concurrent datapath transfers. The TENSOR is a single TPH node with up to 64 nodes capable of mesh-connections in a STAR, RING, hypercube, banyan, or n-ary polygon. As a consequence, up to 64 FPUs can be pipelined and/or be paralleled. Table 1 lists total multiplication operations for a single, double, quadruple, and 64 node STARBURST. All multiplications are complex multiply operations. It is assumed that data has been prestored in favorable order, that the register files are configured as circular buffers and multiple copies of the datasets exist in local memory to each node. Up to 4 nodes, a linear speedup factor of 4 is possible. For a 64-node configuration, the speedup factor is slightly less than 64 because internode traffic is constricted by the 4-port local memory (any more ports could not physically fit on TENSOR). 2M is the width of the data window. N is the number of samples used (resolution is proportional to N, data smoothing is proportional to M). Note, that if M = 64, then the latency of STARBURST is approximately one FPU operation time (or 10 nanosecs) while the latency of a single TENSOR TPH is still only M/4 x 64 = 16 multiply times.

Table 1. DTWD Multiplication Periods on TENSOR (Architecture (TPH)

EQUATIONS (Sec. 2.1.2)	1 NODE	2 NODE	4 NODE (1)	64 NODE STARBURST (2)
14	M/4	M/8	M/16	M/63
15	1	1	1	1
16	N/4	N/8	N/16	N/62
17	1	1	1	1
18	M/4	M/8	N/16	N/62
19	M/4	M/8	N/16	N/62
20	1	1	1	1
21	1	1	1	1
22	N/2	N/4	N/8	N/62
23	N/2	N/4	N/8	N/62

(1) 4 copies of each dataset in local memory
(2) 16 copies in local memory

2.1.3 Singular Value Decomposition

Ewerbring and Luk [Ewer90] describe two applicable SVD methods to the Tensor architecture. One-sided and two-sided Jacobi methods are used on the Connectionist Machine (CM), an SIMD architecture with surprisingly fast throughput. An SVD algorithm is useful for matrix inversion (super-resolution). Simply stated, given a real m x n matrix A, its SVD is

$$A = U \Sigma V^T$$

where U and V are orthogonal matrices and Σ is a diagonal matrix whose elements are the singular values of A. Columns of U and V are left and right singular vectors. In the one-sided Jacobi transformation, an iterative procedure transforms V orthogonally. In the two sided Jacobi transformation, annihilation of two off-diagonal elements of A are executed by two orthogonal transformations on U and V in

$$U^T A V = \Sigma$$

The Jacobi methods should not be directly applied as suggested by Ewerbring because TENSOR in the MIMD mode can outperform CM. TENSOR PEs each have 4 multipliers, a square root and divisor which makes TENSOR at least 4 times faster than CM. In addition, the TENSOR Hyperbar supercedes the CM 12-dimensional hypercube. Also, TENSOR's MIMD nature is best capitalized by using the decomposition approach of Chang and Lee [Chang90]. Here, latency is minimized in unequal pipe delays by a novel buffer assignment on acyclic directed graphs.

The major computational requirements for important real-time processing tasks can be reduced to a common set of basic matrix operations including matrix-vector multiplication, matrix-matrix multiplication and addition, matrix inversion, solution of systems of linear equations, least-squares approximate solution of linear systems, eigensystem solution, generalized eigensystems solution, and singular value decomposition (SVD) of matrices including the generalized singular value decompositions of Paige-Saunders [Paig81]. The first five matrix operations listed above may be computed non-iteratively, and systolic array architectures and algorithms are available.

For example, in the least-squares applications, the main objective is to compute the least-squares residual while the corresponding weight vector often is not of direct interest. We propose a modified version of Kung and Gentleman's systolic QR decomposition array in which the least-squares residual is produced quite simply and directly at every stage without solving the corresponding triangular linear system. This development has a number of significant advantages.

Current research on algorithms and architectures for matrix computation is directed primarily toward the single value decomposition and the symmetric eigensystem problem. The simplest approach to eigensystem computation would be to use the power method, since it requires primarily matrix-vector multiplications. However, this method is not viable because of its severe numerical difficulties for closely spaced eigenvalues, small eigenvalues and their associated eigenvectors. Modern, numerically stable algorithms for the symmetric eigenvalue problem and the singular value decomposition are based on real orthogonal transformations and which are used in the QR algorithm [Wilk65] and the Jacobi algorithm [Schw73]. For the singular value decomposition, one sided orthogonal transformations are used in the Nash-Hestenes method, and a modification of the QR eigensystem algorithm is used in the method of Golub and Reinsch [Golu70]. In trying to find systolic parallelizations of these algorithms, the difficulty is not so much in providing a parallel implementation of the required transformations as in computing the transformation to be used at each stage of the algorithm.

The QR type algorithms proceed in two stages. The first is a preliminary reduction to tridiagonal form for the symmetric eigenvalue problem or a reduction to bidiagonal form for the singular value decomposition. Successive transformations then perform a reduction to diagonal form. This approach has been investigated at Pennsylvania State University [Hell81] and at Stanford University [Schr82]. The principal difficulty is that of performing the reduction to tridiagonal form for the eigensystem problem or to bidiagonal form for the SVD in time proportional to N using about N squared processors! The Jacobi algorithms for the symmetric eigensystem problem and the corresponding one-sided orthogonalization algorithms [Nash79] for the SVD lend themselves readily to vector parallelism [Luk80]. Versions with both matrix and vector parallelism are being studied at Cornell University. The initial method using matrix parallelism attempted to compute all N(N-1)/2 rotations for a sweep in advance, and then perform the rotations [LukF82]. The principal difficulty with this approach is that rotations proceed simultaneously in several planes, and the classical methods and corresponding proofs of convergence are designed for computing optimal or nearly optimal rotation choices in one plane at a time. In effect, some of the rotations are performed using old information to compute the rotation angles. New methods are still needed to understand the convergence behavior of such algorithms. Two alternative architectures avoid the use of old data in computing the rotations. One uses vector parallelism to implement in exact Nash-Hestenes method with a modified order of performing the rotations [Bren82]. The data movements for this architecture are shown in [Spie83]. A second method uses two-sided orthogonalization to permit computation of the SVD in 0(n log n) time using 0(n^2) processors [Bren82].

2.2 Phase Retrieval

Phase retrieval is necessary because the earth's atmosphere destroys the phase of images received from outer space due to diffractions. Feinup's studies suggest a solution by successively approximating the phases with each new image based upon additional apriori knowledge of the events. The algorithm is computationally intensive. It requires 5 TERAOPs of real-time operation. Even using Feinup methods consumes 10-15 minutes of CPU time. A simplified procedure utilizes 300 to 400 Fourier transform pairs on 64 x 64 pixels of image data. The TENSOR architecture can compute 100 64 x 64 FT pairs in 20 millisecs and comes very close to real-time requirements.

Of several algorithms to improve images and obtain super-resolution, procedures by Feinup and others offer clear evidence that correct clear images even though diffracted can be obtainable. The algorithm starts using the region of the diffraction limited image in the surround. This is the known part of the error image. An underlying assumption is that a diffraction limited picture is the sum of a correct image and an error image. First set the support region of the picture to zero. Fourier transform the picture. Next, zero the region of the diffraction domain only where the error region is known with high confidence to be zero. This should be the area in the true spectrum of the correct picture is known. This step is equivalent of a high pass filtering operation. In the next step, perform an inverse Fourier transform and restore the known portion of the error region in the surround leaving the support region untouched. Repeat the above procedure on the complete picture. In the limit this procedure generates the complete unique error picture.

2.2.1 Gerchberg-Papoulis Algorithm

Gerchberg shows that this procedure is an eigenvector construct [Gers89] of the following steps. The iterations or transformations are a linear transformation of matrices of pixel elements. In essence, process each element of a standard unit basis (the pixels) through the algorithmic steps outlined above. The result is a column of a matrix whose partitioned form appears as

$$\begin{vmatrix} A & B \\ Z & I \end{vmatrix}$$

where A is the square matrix (whose rows are the number of pixels in the support). Z is a zero matrix and I is the identity matrix. The total N x N matrix represents the complete picture (surround plus support). One cycle of the algorithm corresponds to a multiplication of the matrix by the current estimate of the error picture. The above matrix must be put into a reduced echelon canonical form. One method is via Guass-Jordan procedures. Other methods are much faster such as LUD for which the TENSOR is finetuned.

In summary, the computational steps (for one iteration) of the above cited constrained iterative algorithm to retrieve phase and obtain super-resolution are

1. Partition the surround and support regions
2. Zero the support region
3. Fourier transform the image
4. Zero the true spectrum area of the correct picture
5. Inverse Fourier transform the image
6. Reinsert the known portion of the error picture

Steps 3 and 5 are standard 2-D FFTs and no special data manipulation nor any complex datapaths are necessary in the architecture. The remaining steps are not simple even though clearing something to zero is basic. Finding the regions and setting up the 2-D address generation in real-time is very complicated. The best technique is to assign each subscript a hardwired address generator. This generator must include an adder (for offset), a counter (for sequential access), and a multiplier (for indexing through columns/rows). Assuming the real-time speeds of 40 MHz clock rates and large frames, a 5 nanosecond 16-bit multiplier and a 100 MHz clocked counter are necessary. The address generator circuits shown in Section 5.3 is designed to handle this ultra fast generation in each TENSOR node. This is only possible with a unique ASIC architecture which was discovered in Phase I to insure that a 64 x 64 pixel frame is corrected in real-time. The Gershberg algorithm extrapolates the spectrum of a band-limited function by extrapolating the error function in the image instead. One important feature of this constrained iterative algorithm is the ability to trade noise amplification for bandwidth. Simply band-limit certain eigenvectors.

2.2.2 Parameter Estimation Algorithm

An alternate approach is to use a parameter estimation technique suggested by Haacke, Liang, and Izen [Haac89]. The partial Fourier transform data reconstruction problem can be simply described as solving for the object function p(x) from the following integral equation:

$$s(k) = \int_{-\infty}^{\infty} p(x)e^{-i2\pi jx}\, dx$$

where s(k) is only available at $k = n \triangle k$ for $n = -N/2,...,N/2-1$. It is well known that this problem is ill posed. Constraints other than data consistency have to be used to obtain a good inversion.

Here, recast the super-resolution problem as one in which the images are clusters of boxcar functions. Then estimate where the true edges are by converting the problem to an all pole model. Use linear prediction theory to estimate the positions of the poles. The trick is to recognize that the derivative of a boxcar is two spikes and the Fourier transform takes advantage of this derivative.

Consequently, the solution to an ill-conditioned set of linear equations is sought. Use an SVD-based least squares procedure. Thus, an SVD operation remains the computational bottleneck. TENSOR is specifically designed to perform SVDs fast because its LU decompositions are done in parallel to generate the eigenvectors and residuals. A parameter estimation approach to super-resolution reconstruction is superior to Gershberg algorithms when images are noisy. Also, systematic modeling errors remain localized in the reconstruction. Hence, targets can be expected to be resolved even when they are spatially close to each other.

2.2.3 Abiss Regularization Algorithm

According to Abiss [Abis88], the mathematical basis for SR techniques are:

1. The Fourier transform (spatial Spectrum) of an object of finite extent is an analytic function.

2. Hence, the entire spectrum can be reconstructed from the segment transmitted by an imaging system with finite aperture.

3. The inverse Fourier transformation then yields the super-resolved image.

4. However, direct implementation of the scheme fails in practice because of atmospheric diffraction noise.

His solution (to step 4) is two-fold:

1. Exploit apriori knowledge of the imaging system and targets.

2. Apply regularization techniques to achieve stability in the super-resolved image.

The ultimate computation is the solution to the following Equation (24).

$$f_a = (A^T A + aI)^{-1} A^T g \qquad (24)$$

This solution can be derived by a singular value decomposition of A. Hence the major computational task of SR is the eigenvalue decomposition task. Numerous optimal architectures exist to speed up this computational sequence. Some of which may have applicability in the current effort. Systolic arrays lend themselves nicely to SVD computations. WARP machines and other vector oriented organizations are also possible. VHLSI may be required. But even these levels of integration may not suffice by 1993 (completion of Phase II) to achieve real-time performance goals.

2.2.4 Arithmetic Fourier Transform (AFT)

The derivation of the AFT which now follows is found in [Boud89]. Here the authors contrast the performance of the AFT against the summation-by-parts (SBP) DFT to determine the error effects of finite wordlength.

Assume that a Fourier Series representation of a real-valued, periodic, function A(t) which has period one, contains no zero-frequency (constant) term, which is band-limited to the Nth harmonic of the fundamental frequency is desired. The Fourier series representation of A(t) is

$$A(t) = \sum_{k=1}^{N} A_k(t) \qquad (25)$$

where each A_k is

$$A_k(t) = a_k\cos(2\pi kt) + b_k\sin(2\pi kt) \qquad (26)$$

where a_k and b_k are the Fourier Series coefficients (FSC) of the kth harmonic term. To help understand the relation between the FT and the AFT, it is useful to consider a bank of N delay-line filters each with input A(t). Each filter line, in essence, relates to the harmonic content of the equally spaced in time samples. The output of the nth filter is denoted by S(n,t) and defined by the formula

$$S(n,t) = 1/n \sum_{m=0}^{n-1} A(t+m/n) \qquad \text{for } n = 1,2,\ldots,N \qquad (27)$$

When we calculate the frequency response of the linear filtering operation of (27), the Fourier series of S(n,t) is then the sum of the harmonics of the nth frequency component of the input signal,

$$S(n,t) = \sum_{m=1}^{N} A_{mn}(t) = \sum_{m=1}^{N} [a_{mn}\cos(2\pi mnt)+b_{mn}\sin(2\pi mnt)] \qquad (28)$$

for n = 1,2,...,N. It is then obvious that the following inverse formula is applicable to compute the kth Fourier harmonic component

$$A_k(t) = \sum_{m=1}^{\lfloor N/k \rfloor} \mu(m)S(mk,t) \qquad \text{for } k=1,2,\ldots,N \qquad (29)$$

where μ is the Mobius function and $\lfloor v \rfloor$ indicates the integer part of v. The AFT calculation in (29) is simple because we can take advantage of the simple unitary values of +1, -1, and 0 for the Mobius apply. Since the input signal is bandlimited, the summation in () only contains from 1 to N terms. The AFT formulation is also deceptively simple because the multiplications in it are trivial. The only multiplications in the AFT are the N-1 scaling factors by 1/n in (27). Since $A_k(0) = a_k$ and $A_k(1/4k) = b_k$ in (26), and hence (29), Bartels shows that one of several possible approaches for obtaining a_k and b_k in (26) is to equate S(n,t) in (27) and (28) and sample it at two different time instants,

$$S_n = S(n,0) = 1/n \sum_{m=0}^{n-1} A(m/n) = \sum_{m=1}^{N} a_{mn} \qquad (30)$$

and

$$T_n = S(n,1/4n) = 1/n \sum_{m=0}^{n-1} A(1/4n+m/n)$$

$$= \sum_{m=1}^{N} (a_{mn}\cos(m(\pi/2))+b_{mn}\sin(m(\pi/2))) \qquad (31)$$

to obtain the AFT relation between the equally spaced samples of the input signal in the left most sum and the harmonically indexed FSC on the right. Now the power of the AFT implementation becomes clear. The Fourier coefficients can be obtained by simple addition or subtraction of the sums S_n and T_n, each of which can be computed in parallel. If the b_k's are not needed, T_n in (31) does not need to be computed. For $k \geq N/2$, (29) simplifies to $A_k(t) = S(k,t)$ and hence $a_k=S_k$ and $b_k=T_k$.

It is useful to point out that the advantages of the AFT algorithm are the following:

1. the arithmetic computations can be performed in parallel channels and are naturally pipelined.

2. only one division per channel is needed.

3. the modularity and simple control structure of the algorithm makes it efficient to implement in VLSI.

4. the algorithm latency time for parallel implementation is relatively short.

5. the computations can be exact; that is, the only possible rounding errors are the one division per channel.

The disadvantage is that approximately $N^2/3$ non-uniformly spaced samples are required whereas an FFT needs only N uniformly spaced samples.

2.2.5 FFT Computations

The key solution to a fast FFT is to partition the dataflow paths onto a device ideally suited for the task. STC has examined the Austek architecture because it is claimed to be a dataflow machine for FFTs. Feinup's phase retrieval task ideally needs 64 x 64 2D FFT with 16 bit precision at the rate of 200 in 20 msec. A study of the Austek device [Aust88] indicates that with two devices one 64 x 64 can be done in 1.638 msec or 320 msec for 200 FFTs.

If 30 chips are used, then 200 FFTs at 64 x 64 can be executed in 32 msec at a cost of \$270 per chip in quantities. This is still cost prohibitive for a Phase III SBIR development into medical electronics. However, some clues to wafer scale integration may be found in this underlying architecture and should be further examined. A block diagram and detailed architecture are depicted in Figures 1 and 2. Note, carefully, the large usage of PALS for the distributive control as is typical of tightly coupled dataflow machines. If wafer scale integration is possible, these control paths must be reusable to save substrate space.

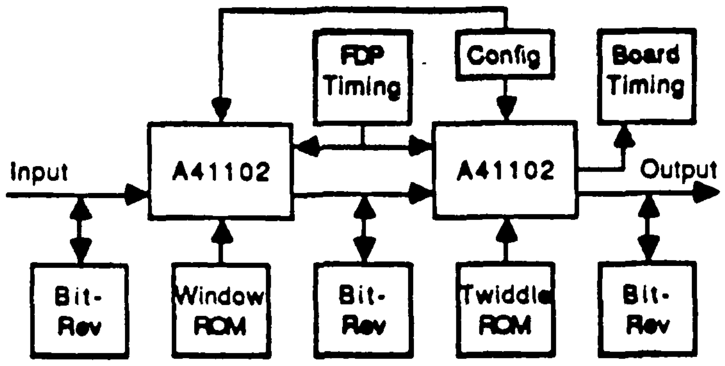


Figure 1. Block Diagram

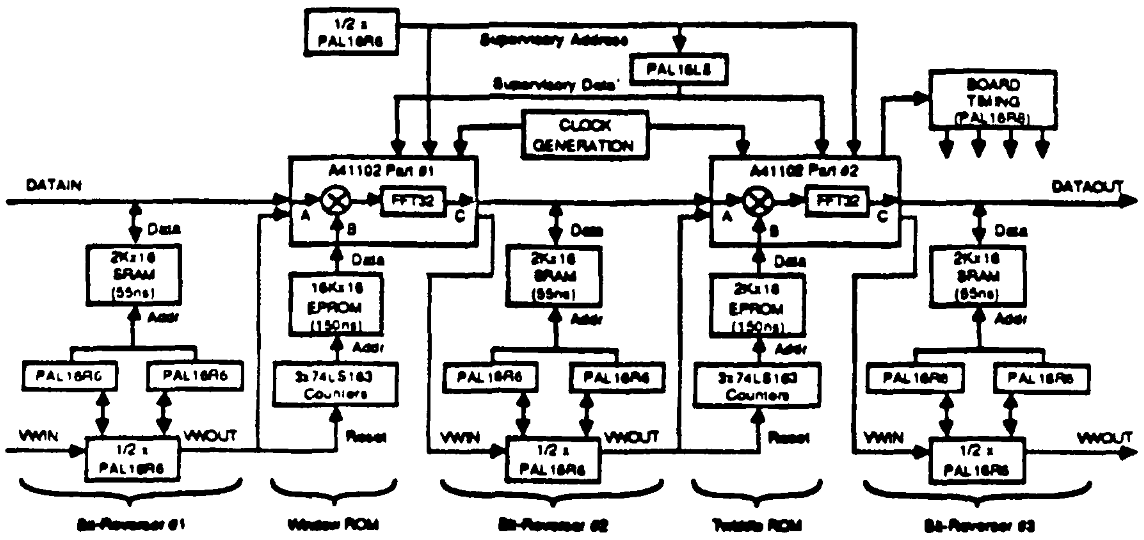


Figure 2. Detailed Architecture

2.2.6 Chirp-Z Transform

Since some sampled data may not be in even powers of two and single line spectra may be desired if the 2D image is sparsely populated with targets, an investigation into the Chirp-Z transform computational sequences was made. Initial observations suggest, however, that a CZT requires 2N log N operations whereas the FFT requires only N log N operations. Hence, the CZT is not competitive. Only when variable resolution and complex data lengths are desired should the CZT be considered.

Some important observations about the CZT are:

1. The results of the CZT are mathematically identical to those produced by the FFT and DFT.

2. The resolution of the CZT is determined by the length of the data sequence. Windowing isn't necessary as in FFT where resolution and spectrum spreading will result.

3. The CZT can operate on a data sequence of any length.

4. If carefully parallelized, the CZT may be as fast as the FFT.

5. The CZT uses the same amount of processor memory as the FFT.

2.2.7 Critical Algorithms for Superresolution

It is commonly known that high computation rates are necessary for SDI signal processing applications. Because of this need, much research has been done on systolic, parallel and pipelined processing. Several algorithms have emerged as important. Our development focused on applying the optimal forms of these algorithms to our proposed array structure. The kernel "signal process" in most cases is of the form y = Ax+B (a multiply-accumulate step).

Hence, Faddeev's algorithm attempts to find **C X + D**, given **C**, **D** and equation **A X + B** where **A** is of full rank. Select suitable linear combinations of the rows from **A** and **B** to add to the row below the line so that only zeroes appear in the lower left hand quadrant. The desired result, **C X + D**, will appear in the lower right hand quadrant [NaPH86]. Since the only requirement is that **C** be annulled, ordinary Gaussian elimination can be used instead of backsubstitution. Gaussian elimination does not guarantee numerical stability, so a QR orthogonal factorization using Givens rotations is performed. The process given in [NaPH86] also requires that the only restriction on **A** is that it is full rank. As given, Faddeev's algorithm can solve general matrix factorization problems including inner and outer products, triangular system solution, matrix factorization (LU and QR), matrix manipulations (mult., inverse, add, transpose), linear systems solutions, banded matrix system solutions, and full rank LS systems including constrained and generalized cases. Our study of Faddeev's algorithm focused on an efficient implementation on the general systolic array to compute the matrix factorization problems above.

Total Least Squares (TLS), as originally developed by Golub and Van Loan [GoVa83], is a method for solving the system of equations **A X + B** when both **A** and **B** are corrupted. Conventional least squares techniques only deal with errors in **B**. The general TLS technique attempts to find the "best" ndimensional subspace approximating the range {[**A** **B**]} instead of forcing range {[**A** **B**]} to be the same as range {**A**}. The TLS technique was recently extended [Zol87] to provide a method for the case of an arbitrary number of "right hand sides." Also, a closed form solution for the minimum norm solution associated with the rank deficient problem was derived. Applications of TLS to source covariance matrix estimation and minimum variance distortionless response beamforming were presented. Our study of desirable signal processing algorithms considered TLS techniques so that algorithm mapping is complete.

Singular Value Decomposition (SVD) is an effective method of matrix rank determination, data compression, and matrix (pseudo) inversion. Since the development of a systolic architecture implementation for the SVD by Brent and Luk [BrLu82, BrLV83] research has continued allowing implementation in a variety of structures [Ips84, MoLa86, FeHa86]. The most computationally efficient method for calculating the SVD of an m x n matrix **A**, m > n, appears to be an initial QR decomposition followed by the SVD of **R**. Both decomposition techniques use plane rotations to reduce the input matrices.

Because plane rotations may require involved division and square root procedures, improved algorithms [BaIp85] were explored. The important engineering trade-offs appear to be the number of specialized operations required (divisions and square roots) versus the complexity of otherwise necessary scaling computations. The Fast Givens algorithm computes an extra division instead of the scaling (by use of shifts) in the Scaled Givens algorithm [BaIp85]. Our research involves a study of the required trade-offs to determine which algorithm is more attractive.

3.0 Architecture Analysis

Although the selected architecture had to efficiently compute the algorithms cited in Section 2, additional design parameters were used, one of those considers the overall flowthrough of tasks. The following equation was used to measure the overall performance of the machine and takes into consideration "Amdahl's law". Amdahl's law considers the typical realities of an algorithmic specific architecture. It reflects how much actual concurrency is gleaned by parallel processors versus actual performance constrained to only a few processors. The simplified equation [Lund87] measures the reduced performance on certain steps in the computational sequence quantitatively and is

$$T = f_s * F/R_p + f_p * F/(P * R_p * U)$$

$$= \text{elapsed time}$$

where

F - total number of floating point operations to be evaluated.

R_p - the number of floating point operations per second for one processor.

P - the number of processors.

f_s - the fraction of floating point operations which must be executed in a serial mode (only one processor being utilized).

f_p - the fraction of floating point operations executed with all processors executing.

U - the average utilization during the execution of the sections of code attempting to utilize all processing resources.

The effective system computation rate can then be given by the following equation:

$$R_s = F/T = \text{System computation rate (floating point operations/second)}$$

The bulk of primitive processing for all the algorithms except super-resolution was vector inner products and the efficiently fast management of the large vectors. The super-resolution algorithm needed a fast matrix inversion. This single computational step remains highly recursive. Hence, data dependencies effectively slowed down conventional processing schemes. In effect, a new iteration in the inversion has to wait for a result from the previous input-dependent computation. For single or few computational engines, only a low degree of multiplexing of ALUs would result. Architects naturally conclude that many

PEs help. And systolic or massively parallel organizations do speed up such problems. The cost is then transferred into the complexity of data stream management. Matrices must be remapped onto the memory space dynamically, somewhat like bit shuffling in the FFT. In this case, the algorithms have to be extensively transformed into regular and repeatable steps so that the data is dispersed onto a grid of PEs. Then, an orderly progression of input and output streams appear. Latency can be high, however. Yet, modularity of design, high throughput, and minimal storage requirements at the local PE remain attractive to many systolic or other array organizations.

Given the need to solve both vector as well as matrix data at ultra high speeds, a first choice would be also a combination of dedicated DSP devices and fast floating point processors (FPUs). That architectural choice did not suffice because esoteric constraints like board space, edge connections, and VLSI could not support fast computations. A better solution is a regular array of PEs that can each remain autonomous when needed. The autonomy is often useful for the scalar operations and ancillary operations which clean up fragmented processes. These free standing processors also function as data managers where corner turning of matrix streams is necessary.

Data memory system design is critical to speed and throughput when only a finite space is anticipated. For the SDI tasks described earlier data memory with a local 1 mword space can store typical vector and matrices. But a balance must be made between the local data memory and the shared data memory. Local data memory must be fast, equal to if not comparable to that of ALU speeds. But such high speed memory is very costly. Economic costs then force us to consider slower main memory rather than total cache. Cache memory is best invoked when "locality of reference" is strong. Otherwise, the cost of cache is excessively high. Caching data is not simple. Major computer manufacturers invoke exotic cache data management to prevent cache misses. Cache coherency problems overwhelm must multiple processor shared memory organizations. (Cache coherency means that cache contents in one processor are useless because of a write on another processor to an exact copy of the line in both caches.) Hence, cache is not critical to the SDI tasks since data is expected to arrive very regularly anyway.

3.1 Very Wide Word Machine (VWWM)

Examining all architectural prospects for the three specific SDI tasks suggests that at the minimum a multiple ALU PE level architecture is optimal. The actual PE described in detail in Section 5 is a 4 FPU, 4-port local memory configuration with a 64 word register file shared by all FPUs. The organization of each PE is extremely flexible in that full datapath interconnect between any of the internal PE resources (cache, n-port memory, FPUs, and register file) is possible. Further, all paths are simultaneously available. Because this very flexible crossbar provides such versatility, the control word for each PE is very large, exceeding 300 bits of micro-instruction. However, such wide control also enables full concurrency at the firmware or machine level.

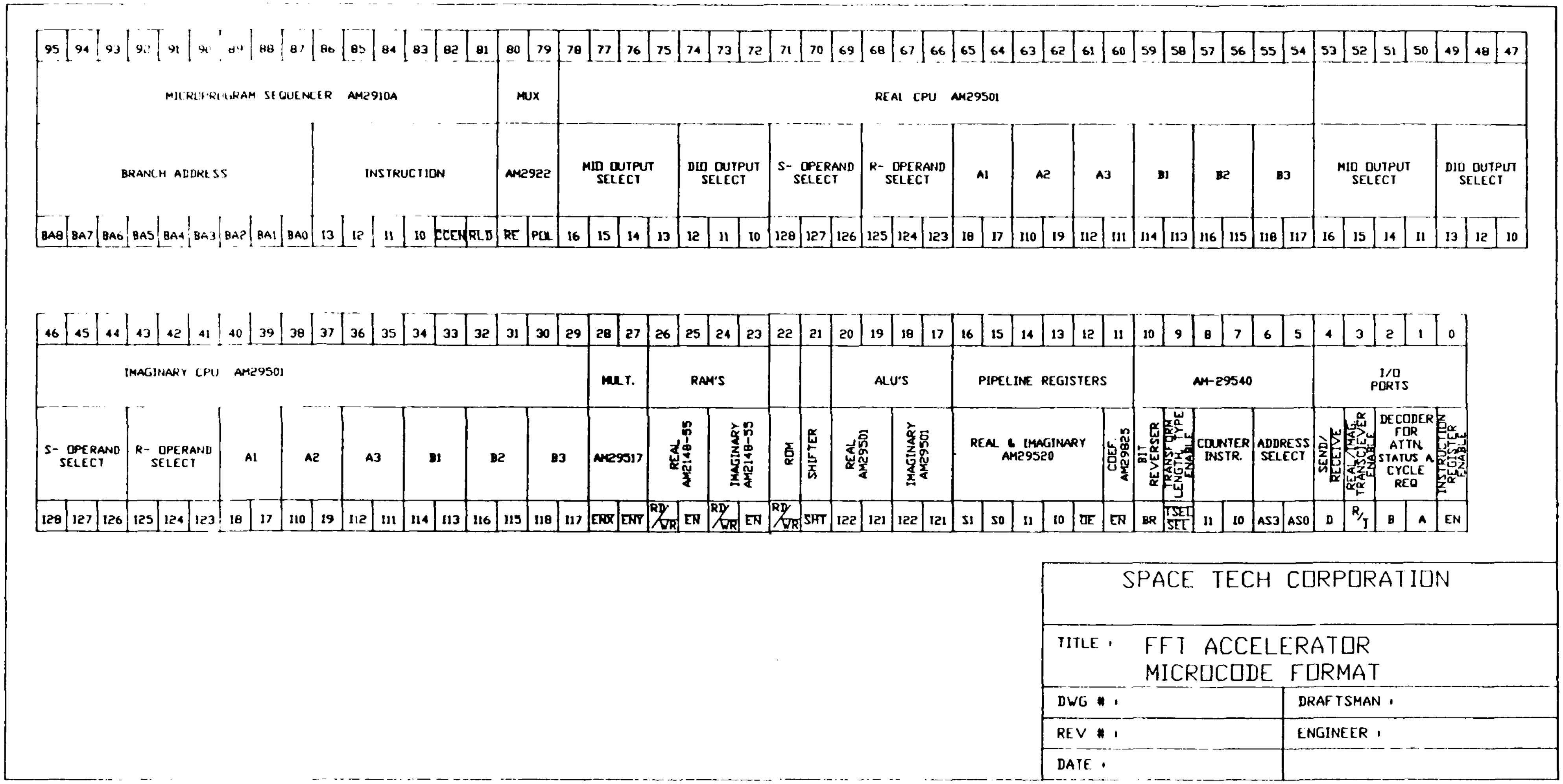
The actual microinstruction layout (only for the FFT accelerator is shown in Figure 3. Several micro-order fields, all of which are independent make control fully parallel so that SIMD and MIMD operation can be supported. In addition, the microprogram control allows for literals at the microcode level so that fast transfer of coefficients and constants is supported (vital to the Hilbert, arithmetic, and Fourier transforms). Each PE comprised of the 4 FPUs and supporting memory has individual microprogrammable control units resident locally to them. This very wide microword machine can therefore execute 4 multiplies concurrently. A typical microinstruction format is attached in a D-size schematic (247 bits are required for just 2 FPUs). The reason for organizing the PE around 4 multipliers is that although complex data multiplication of

$$(A + jB) * (C + jD)$$

can be accomplished by noting that the expression is equivalent to

$$[(A - B)D + A(C - D)] \\ + [(A - B)D + B(C + D)]$$

which is 3 multipliers, an FFT butterfly operating on complex data can effectively use 4 multipliers. Even better performance is achievable if tricks like invoking surrogates is used [Conn89]. Another important reason for a 4 multiplier PE is the fast execution of row/column reduction in the LU decompositions.



95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47
MICROPROGRAM SEQUENCER AM2910A															MUX		REAL CPU AM29501																															
BRANCH ADDRESS									INSTRUCTION						AM2922		MID OUTPUT SELECT				DIO OUTPUT SELECT			S- OPERAND SELECT			R- OPERAND SELECT			A1		A2		A3		B1		B2		B3		MID OUTPUT SELECT				DIO OUTPUT SELECT		
BA8	BA7	BA6	BA5	BA4	BA3	BA2	BA1	BA0	I3	I2	I1	I0	$\overline{CCEN}$	$\overline{RLD}$	$\overline{RE}$	$\overline{PL}$	I6	I5	I4	I3	I2	I1	I0	I28	I27	I26	I25	I24	I23	I8	I7	I10	I9	I12	I11	I14	I13	I16	I15	I18	I17	I6	I5	I4	I1	I3	I2	I0

46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IMAGINARY CPU AM29501																		MULT.		RAM'S						ALU'S				PIPELINE REGISTERS						AM-29540						I/O PORTS				
S- OPERAND SELECT			R- OPERAND SELECT			A1		A2		A3		B1		B2		B3		AM29517		REAL AM2148-55		IMAGINARY AM2148-55		ROM	SHIFTER	REAL AM29501		IMAGINARY AM29501		REAL & IMAGINARY AM29520					COEF. AM29825	BIT REVERSER	TRANSFORM LENGTH, TYPE ENABLE	COUNTER INSTR.		ADDRESS SELECT		SEND/ $\overline{RECEIVE}$	REAL/IMAG TRANSCEIVER ENABLE	DECODER FOR ATTN, STATUS A CYCLE REQ		INSTRUCTION REGISTER ENABLE
I28	I27	I26	I25	I24	I23	I8	I7	I10	I9	I12	I11	I14	I13	I16	I15	I18	I17	$\overline{ENX}$	$\overline{ENY}$	RD/$\overline{WR}$	EN	RD/$\overline{WR}$	EN	RD/$\overline{WR}$	SHT	I22	I21	I22	I21	S1	S0	I1	I0	$\overline{OE}$	EN	BR	TSEL SEL	I1	I0	AS3	AS0	D	R/T	B	A	EN

SPACE TECH CORPORATION

TITLE : FFT ACCELERATOR MICROCODE FORMAT	
DWG # :	DRAFTSMAN :
REV # :	ENGINEER :
DATE :	

Figure 3. FFT Accelerator Microinstruction Format

3.2 Control Architecture

The control unit of each PE is configured around a writable control store to allow user support for microprogramming and minimize slow EPROM usage. The control unit is comprised of a RAM control memory, microsequencer, micro-interrupt controller, microstack for context switching at the microprogram level, and conditional microbranching to allow hardwired conditional jumps for overflow, underflow, etc. A central controller, also microprogrammable, maintains synchronization across several PEs and manages dataflow in multiple PE organizations. This distributed control strategy increases the speed of the overall architecture while supporting free standing PE operation (useful for scalar operations). The control unit shown in Figure 4 contains the sequencer control for conditional and unconditional jumps and branches, indexing, I/O function control, and bootstrap microroutine. The microprogram sequencer must operate at 50 nsec. The total delay of the circuit shown using 1990 technology is 43 nsecs. Conservatively forecasting 1992 devices suggests that this is a low risk development.

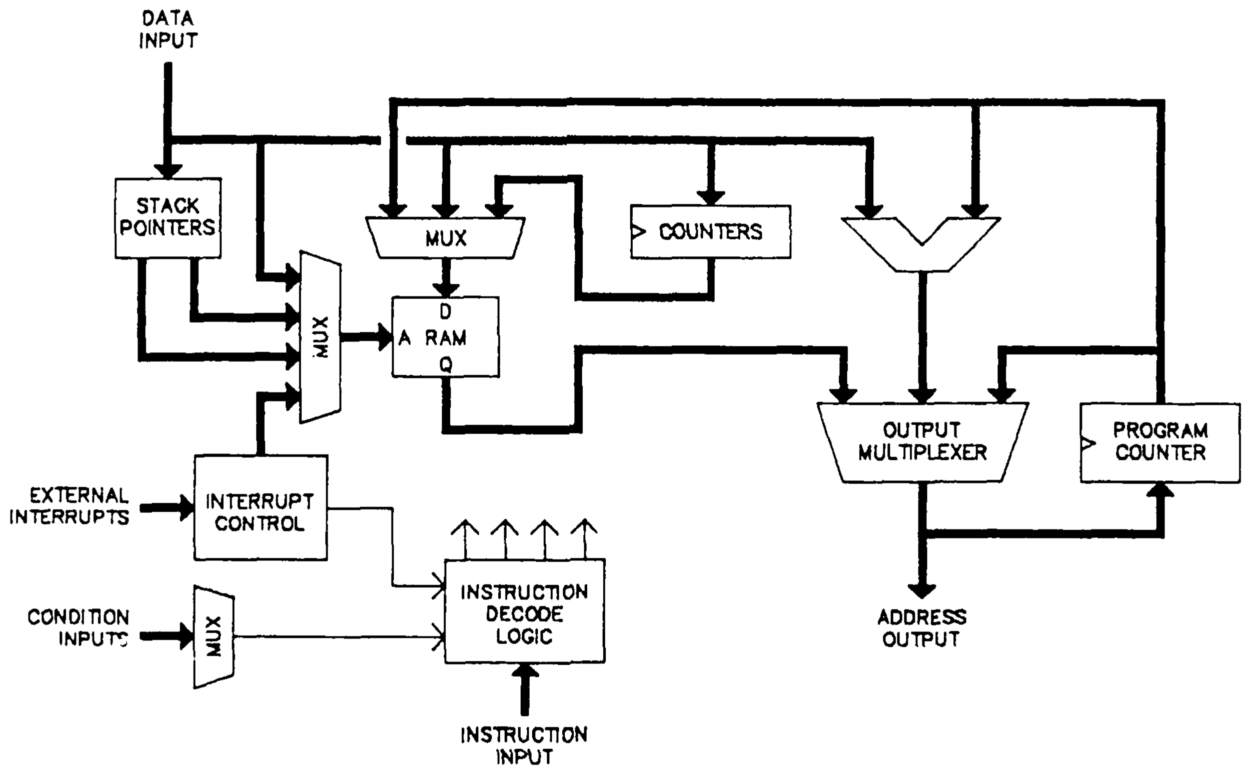


Figure 4. Microprogram Sequencer

The design is markedly different from any commercial control unit available. First, several levels of interruption are supported. Second, relative micro-addressing is handled. Third, micro-index addressing and pre- and post- increment/decrement of stack micro-pointers can be realized, all directly in hardware.

The microinstruction format for controlling a PE is depicted in an attached D-size drawing. Here, 247 control bits are required in 22 fully independent fields so that full concurrency is achieved. Each of the 22 fields controls major hardware resources of the PEs. For example, the microprogram sequencer field is 17 bits wide, with 11 bits assigned to the branch address and loop counter, and 6 bits assigned to the sequencer instruction field to control unconditional jumps, etc. This format must be extended further when the ASIC crossbar and 2 additional multipliers are added to TENSOR.

3.3 Interconnection Architecture

The optimal PE interconnections are those which minimize data pre-shuffles and maximize throughput. Two non-competing data organizations can be found in the SDI algorithms, vectors and matrices (1-D and 2-D, respectively). In effect, PEs need to be pipelined (Wigner ops) or paralleled (SVD). With sufficient local memory, no complicated interconnections are needed. For the operations of convolution, correlation, filtering, and the Hilbert transform, arrange the PEs in stripline or linear fashion and perform cross multiplications as needed. For the matrix operations, organize the data stream so that matrix products occur in parallel. The PEs can then be interconnected in a 2-D regular grid (for systolic operation). Any regular mesh will suffice. Hence, from the need to support both vector and matrix manipulations, a reconfigurable mesh connected organization is best, similar to the Connectionist machine but not exactly the same. The CM is an SIMD machine only, while the architecture of this study is capable of SIMD and MIMD operation. TENSOR, additionally, has 4 ALU pipes.

4.0 Technology Assessment

4.1 GaAs

In accordance with the proposed technical effort, it was suggested that this effort consider certain technologies for integration. Researchers at Matsushita recently discovered that the above cited integration is now technically feasible. The propagation delay of this new circuit stays at less than 200 ps up to a maximum load capacitance of 0.7 pF. Also, this gate is at least three times faster than the BFL gate with a fanout of 7 and dissipates one sixth the power of an ECL gate made exclusively of HBTs. Delay versus power is shown in Figure 5.

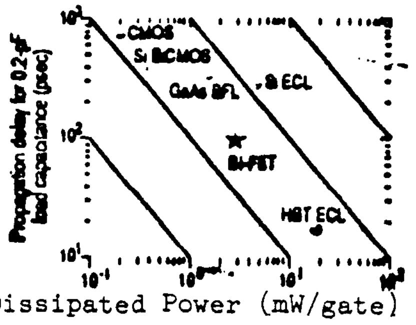


Figure 5. Propagation Delay Versus Power for BiFET Devices [Mokh89]

Current multivalued CMOS circuits are typically not fast. Worse yet, they have low noise margins. (Japanese researchers have just reported on improved CMOS MVL circuits which we discuss elsewhere.) GaAs devices, especially HEMT, are very attractive for multivalue logic because switching values can easily go rail to rail, balanced encodings are possible and more importantly, fan-in and -out are facilitated by equal and output impedances. These GaAs features are directly beneficial to systolic array realizations where regular cell structures are important.

A very important technology now available is GaAs-on-Silicon. Here, the frequency of lattice dislocations is lower and more uniform so that systolic floorplans are easier to generate with more confidence. Furthermore, this technology supports emitter-down HBTs, a natural for GaAs in MESFET realizations. MIT, TI and Japanese researchers substantiate the design methods to be followed in TENSOR. Lastly, this approach lends itself nicely to unified optical and electronic signaling on the same substrate!

Gallium Arsenide often has advantages over similar silicon realizations. According to Trout and Givone [TrGi79] and Upadhyayula [Upa80], GaAs MESFETS may be viable as variable-threshold devices. Jensen, Larson and Waldner [JeLW] discuss potentials for GaAs technology. White [Whi87] presents a topology for a GaAs FFT building block capable of performing a 16-point complex transform in 160 ns or less. He states that the GaAs niche for the FFT appears to be very high speed transforms of modest size.

4.2 Electro Optical Interconnects

Because discrete electronic logic operates efficiently at 10 GTbit/sec rates and optical interconnects have very wide bandwidths, low loss and low crosstalk, total system throughput at teraop speeds is best achieved by marrying these two technologies. An intimate fusion is inevitable. HYPERBAR is an optical wiring for wideband data and control interconnections. It seeks to exploit the 'zero time skew' characteristics of fiber optic wiring (and thus ensure data/clock synchronism). (The transit time of light in 1cm of space is about 33ps, which can handle 10^{12} bits/sec with almost zero clock skew). HYPERBAR invokes an optical 'wiring harness' with conventional bulk optic elements, lenses, prisms (all within current SOA). All key devices have already been demonstrated. Hence, the combination of our TENSOR architecture (pipeline processor approach) and a perfect-shuffle interconnection (optics) enhances matrix addressing, control, timing (solid synchronization) to produce a system of unparallelled power and versatility in STARBURST.

The optic highways function as an NxN crossbar using electrically controlled exchange bypass modules (EBMs). (The electronic equivalent is shown in Figure 6.) Lithium niobate directional couplers can be used herein. E/O interface is best with a **pin** detector with separate electronic gain. Such a detector should be fabricated with III-V based technology formed with MQW material. This material has a band edge so electrical tunability (electroabsorption) is possible. High speed switching is now possible with modulation. "Reading" remotely thus doesn't need a hot optical source close to the electronics. This helps timing and control at picosecond rates. Because such modulators are optically nonresonant, they are fortuitously insensitive to temperature and wavelength.

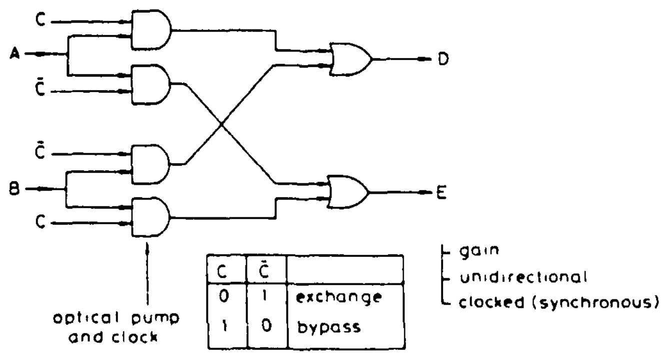


Figure 6. Electronic Equivalent of EBM [Midw87]

4.3 Chip Level Integration

Much of the TENSOR architecture at the PE chip level is VLSI. Thus, increasing densities will profoundly affect ultimate throughput. In this proposed architecture, Space Tech anticipates that chip densities equivalent to the BIT

3130 FPU will be possible by third quarter 1990. A 25% increase in density is conservatively possible by 1992. However, this is insignificant to the performance specified currently. Unless a new technology like BiCMOS or GaAs develops commercially to the VLSI levels necessary, the architecture of this study should not significantly change in the next 5 years. Major gains will be possible in the custom or ASIC device yields and this design fully exploits this opportunity. The TENSOR PE organization is ideally suited to increasing chip densities and their integration at the board level with no change to the architecture. This architecture constancy is vital to the software tools which will be generated by the Phase III codeveloper (compilers, assemblers, OS).

Anticipating device level integrations in 1992 is easier when the 1988, 89, and 90 levels are examined. In 1988, a fast 16-bit FPU became available. In 1989, the high speed BIT 2110 and 2120 appeared. Industry projections for the 4th quarter of 1990 indicate that the 64-bit FPU shown in Figure 7 will be available. This device is similar to the 3130 FPU and constitutes the essential requirements of each FPU for the PEs.

4.4 Board Level Integration

Each PE is a true 64-bit fully parallel machine with 4 64-bit FPUs each of which contains a 64 bit multiplier, adder, divider, and square rooter. To minimize edge connections and keep boards small, significant functionality must be available in each chip utilized on the boards. There is no extra space. To meet such board level integration requirements, the random glue logic cannot be implemented with discrete devices like 7474s any longer. The TENSOR architecture maximizes the incorporation of the VLSI glue logic. The crossbar circuit is one of several innovations which TENSOR captures. The multi-dimensional address generator incorporates the same crossbar, thus achieving higher board densities. Local memory is 4-port on the chip directly.

5.0 The TENSOR Solution

The major technological challenge at the node (or CPU) level is not the selection of the FPUs or fast local memory, etc., but the organization of and the minimal path delays therein. Architects spend considerable time balancing data delays. A major achievement towards this solution has been made in Phase I. The preliminary Phase I studies indicate that the tensor design should consist of the Tensor Processing Hardware (TPH) with fast hardware address generation and data transfer critical to CSO resolution, spin rate ID, and phase retrieval. The main thrust of future Phase III development for the architecture is to organize it into efficiently coupled multi-node TPH modules connected via a HYPERBAR, providing massively parallel execution.

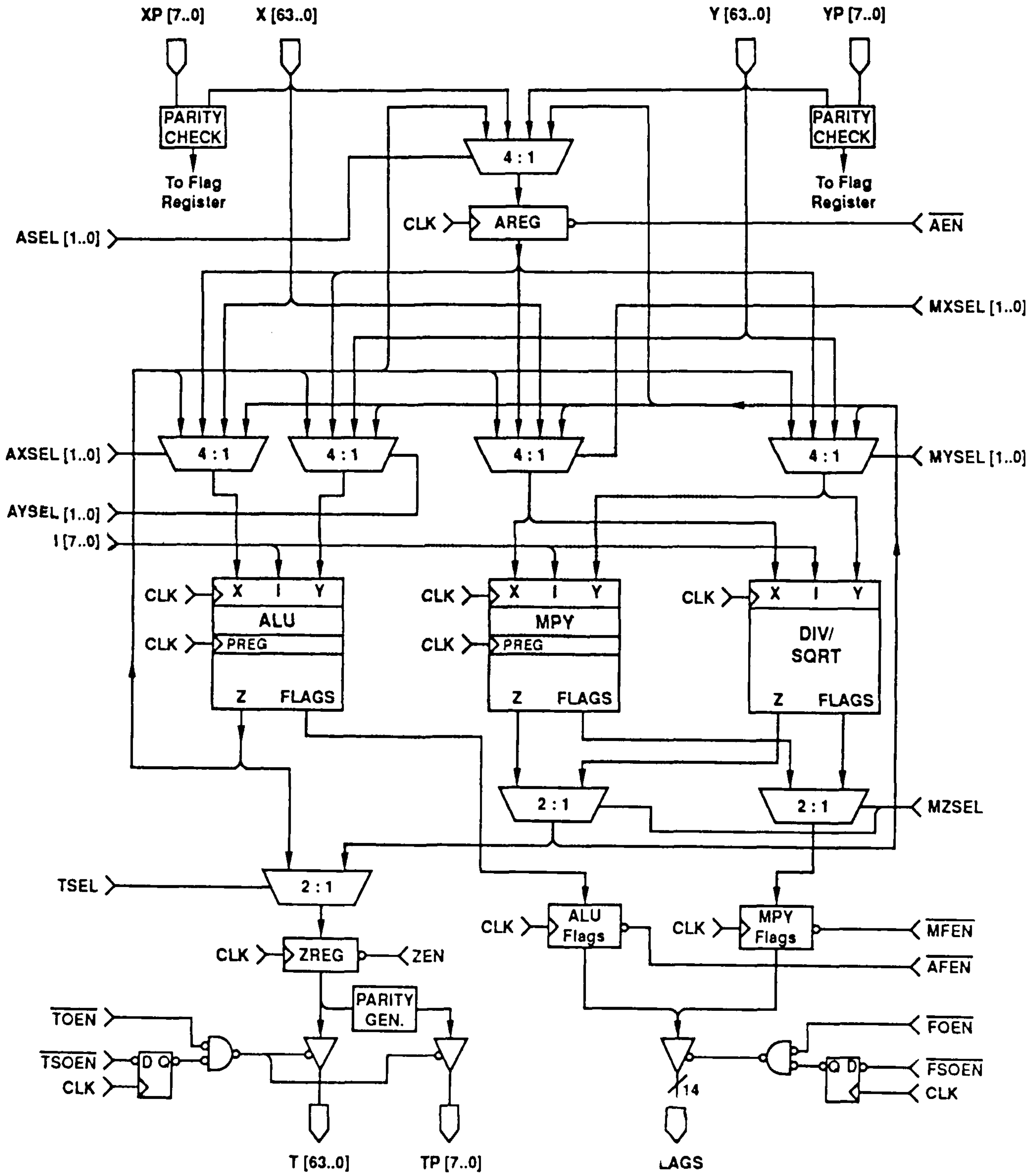


Figure 7. FPU Organization

5.1 TPH Processor Engine

The design parameters for the single TPH node were chosen to enable high throughput with **either pipeline or parallel** data processors, each PE containing 4 multipliers, adders, and square root/dividers. Separate address processors provide indexing capability for vector, matrix, and tensor opera-tions, thereby freeing the data processor for total arithmetic operations. An auxiliary memory contains the coefficients for the Hilbert and Fourier transforms. Four banks of local memory permit continuous processor operation while I/O is active. Hence, true overlapped I/O is supported. Hardware specifications include

ALU

- 4 hardware multipliers, adders, dividers, square root
- 10 nsec 64-bit products
- 25 nsec 32-bit quotients
- 32- or 64-bit wordlengths
- IEEE STD.754 floating point
- 4 pipes or 4 parallel ALUs per PE
- Full XBAR configurable

MEMORY

- Local 4 banks of 16k x 64-bit, 20 nsec access
- 16k x 64 user-constants memory (expandable)
- 4k x 350-bits RAM writable control store
- 64 mbytes main memory directly addressable
- 64 x 64 register file local to 4 ALUs serving FIFOs, Circular buffer, stacks
- 2k x 60-bit EPROM for bootstrap and I/O op

The TPH node supports parallel, vector, and systolic interconnects to other TPHs, where the number of modules required is determined by the matrix size. Data formats of 16-, 32-, 48-, and 64-bit fixed-point, as well as 32- and 64-bit floating-point representations are allowed. To increase the usability of the TPH design, each module will have the same PE architecture, so that both an SIMD and a MIMD machine can be configured with little remicroprogramming. The objective of this hardware replication is to allow the system flexibility to increase its precision and dynamic range.

The TPH hardware can be considered as one PE in the STARBURST system. It is comprised of 4 ALUs each equivalent to the processing power of the BITt 3130, a 10 nsec dataflow 64-bit floating point unit. More importantly, internal feedback of each FPU parallelizes ALU, MPY, and DIV/SQ operations. Now, for the first time, row/column rotations are directly achievable in the same PE cell so that only static data rather than costly data shuffling is needed. 4

BIT chips are to be used for a total TPH throughput of 800 mflops. The proposed architecture appears in Figures 8-10.

The processor unit also has a cache and auxiliary memory, coefficient table (FFTs, Hilbert), and a powerful crossbar. This crossbar has 12 independent and fully parallel 64-bit sets of paths so that not only 12 way but also 12 concurrent paths may be taken in 10 nsecs. This phenomenal design represents significant innovation in an ASIC device. Space Tech expects to file for patents, herein. Because of fully flexible configuration, the computationally intensive matrix inversion and WV algorithms are now computationally trivial. Hence, at the TPH level, 800 mflops are possible. Connecting 8 TPHs with the HYPERBAR provides 6.4 gflops. A single STARBURST with 16 HYPERBARs provides 100 gflops. Ten STARBURSTS support one TERAOP.

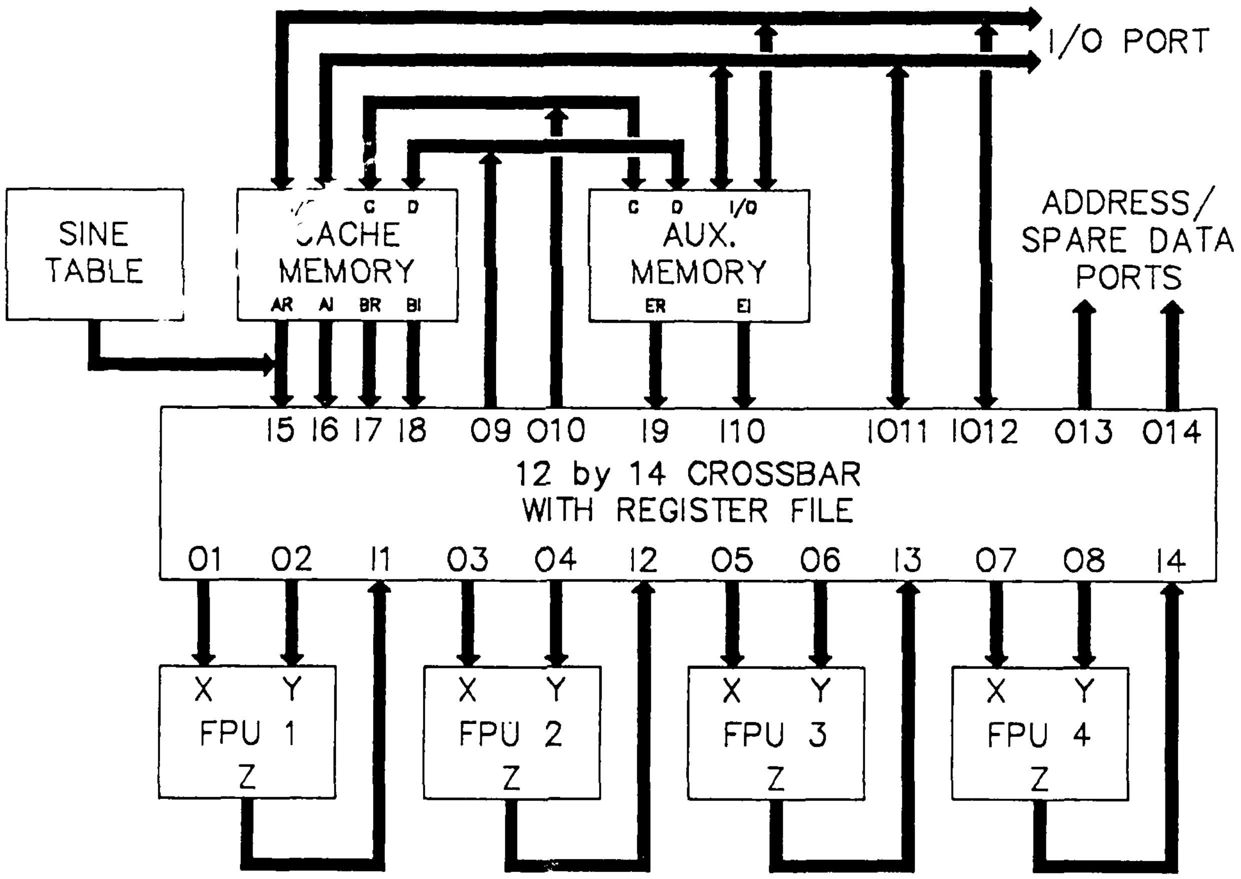

I/O PORT
SINE TABLE
C D
CACHE MEMORY
AR AI BR BI
C D I/O
AUX. MEMORY
ER EI
ADDRESS/ SPARE DATA PORTS
I5 I6 I7 I8 O9 O10 I9 I10 IO11 IO12 O13 O14
12 by 14 CROSSBAR WITH REGISTER FILE
O1 O2 I1 O3 O4 I2 O5 O6 I3 O7 O8 I4
X Y FPU 1 Z
X Y FPU 2 Z
X Y FPU 3 Z
X Y FPU 4 Z

Figure 8. Tensor Processor Hardware

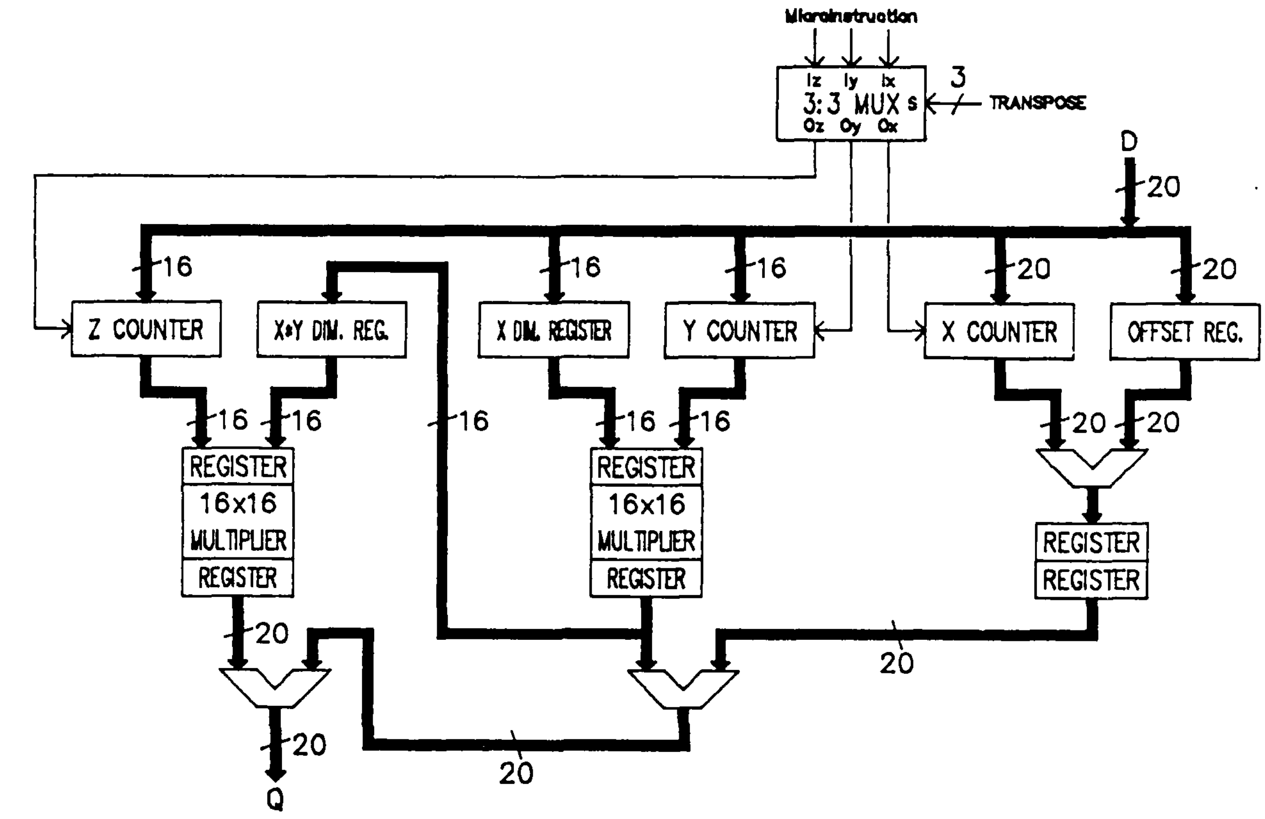


Figure 9. Three Dimensional Counter

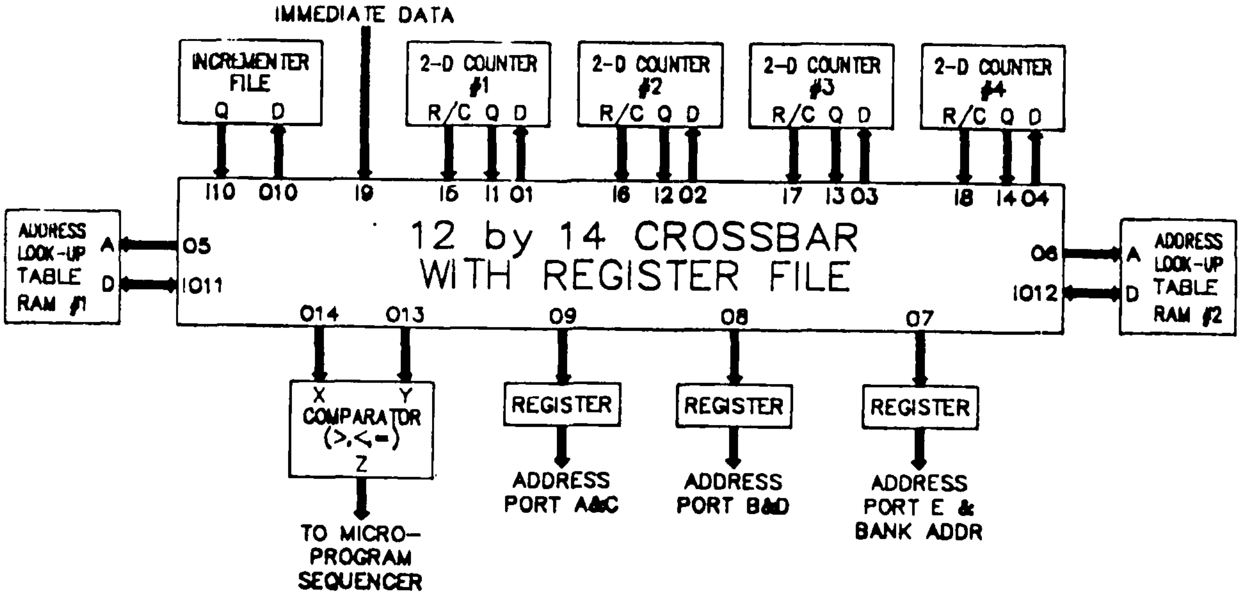


Figure 10. Address Generator

5.2 N-Port Local Memory

Critical to the efficient and speedy implementations of the PEs is a very dense multi-port memory for the 4 FPUs. Access time of such a device needs to be in the order of 10 nsecs for fully parallel 64-bit words. At least 4 ports are necessary to handle two complex data storage paths (real and imaginary) and assist in complex multiplication in the SVD operations. The device must also be a true **independent** 4 port device and not multiplexed as current versions have been. For the first time in 1990, a commercial version of a 4-port chip was made available at low prices. A schematic of the required configuration for the TENSOR machine is shown in Figure 11.

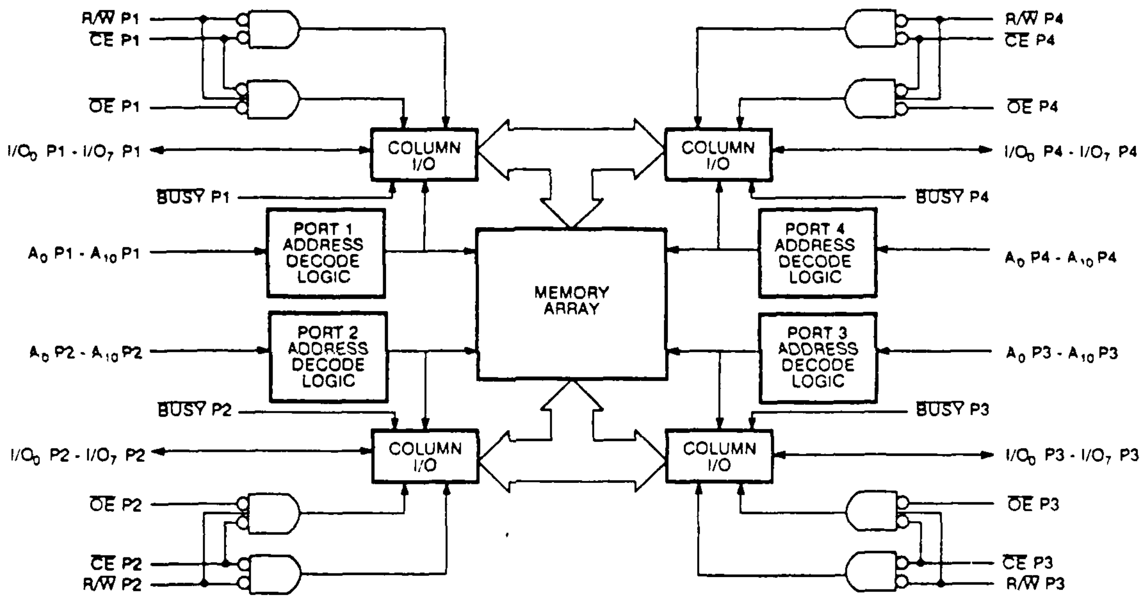


Figure 11. 4-Port Local Memory Architecture

5.3 N-Dimensional Address Generator

Algorithmic speed at the TPH level is enhanced by the same XBAR in the hardwired N-dimensional address generator. This circuit provides the throughput acceleration unreachable otherwise. The address generator uses sets of two-dimensional counters as configured in Figure 10, also hardwired directly. With this uncommon hardware, the TPH can address a single random element in an n-dimensional matrix, a row incrementing or decrementing through it, a column incrementing or decrementing through it, a main diagonal, a minor diagonal, and n combinations of the above simultaneously. The address generator also represents innovative technology which Space Tech will file patents on. This ultrafast addressor races through the SVD, LUD, GE with and without pivoting, Fadeeva inversion, Toeplitz manipulations, and numerous other matrix intensive algorithms.

5.4 ASIC Crossbar

An innovative circuit to each PE is a 12 port crossbar with register file and 2 bidirectional I/O ports. The actual design is shown in several schematics attached to this report. The crossbar is unusual in that 12 fully independent paths are available internal to each PE to allow all four FPUs to transmit and receive 64-bit data in **parallel and pipelined fashion!** To keep board level density low and minimize edge connections, a register file embedded in the crossbar device is accessible by all FPUs. This "shared " temporary scratch pad space is the only high speed local memory needed by the SDI algorithms. The crossbar is a ultra high speed ASIC design completed in joint development with ILSI. This uniquely innovative design is patentable and increases the speed of the 4 FPU PE by orders of magnitudes. Schematics are included (under separate cover) for each functional module of the 12 x 14 crossbar. Block diagrams of each functional module are depicted in Figures 12-15.

With such a crossbar, each PE in TENSOR can be dynamically reconfigured to perform real and complex multiplication and accumulation of 64-bit words in real-time. The crossbar designed can support several PE arrangements some of which are shown in Figure 16. The reconfiguration is transparent to the user. All reconnects are controlled at the microcode level in microinstructions. The interconnects expedite the direct and hardwired execution of the critical multiply/accumulate kernel operation

$$A \times B + C$$

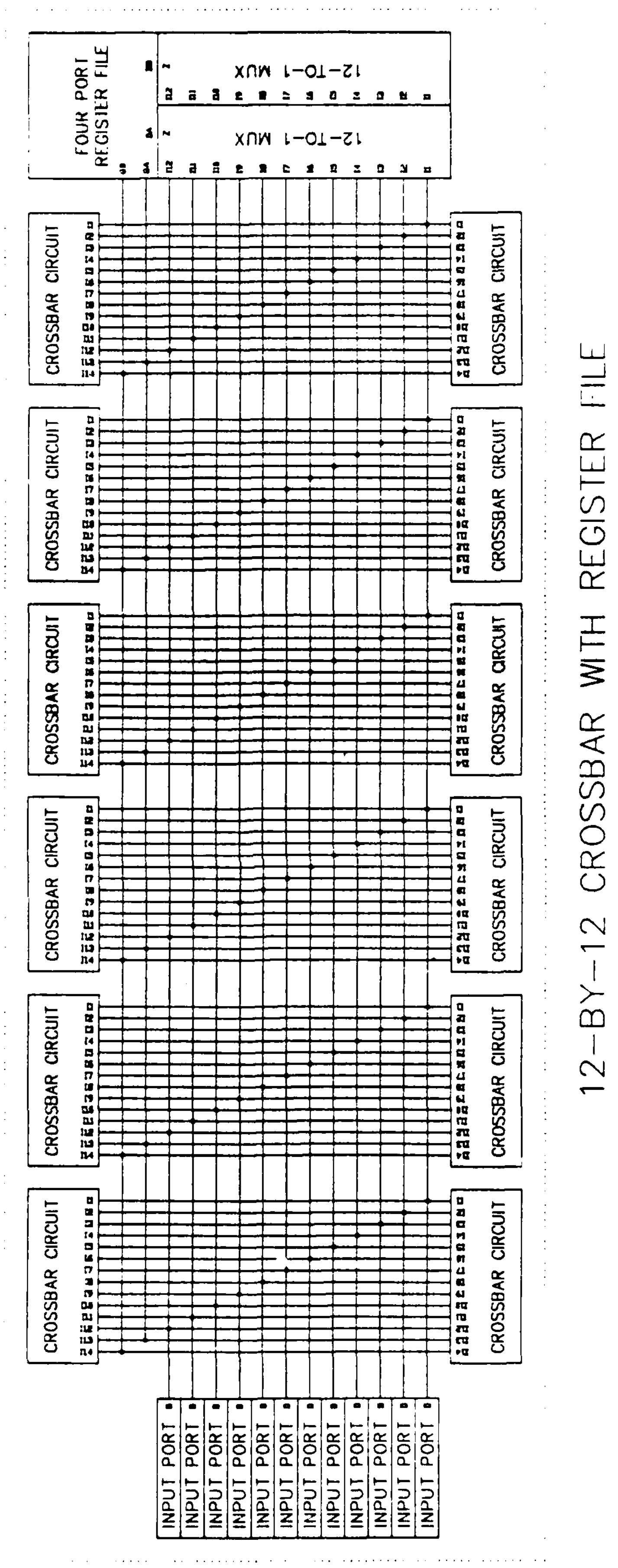


Figure 12. Crossbar Module

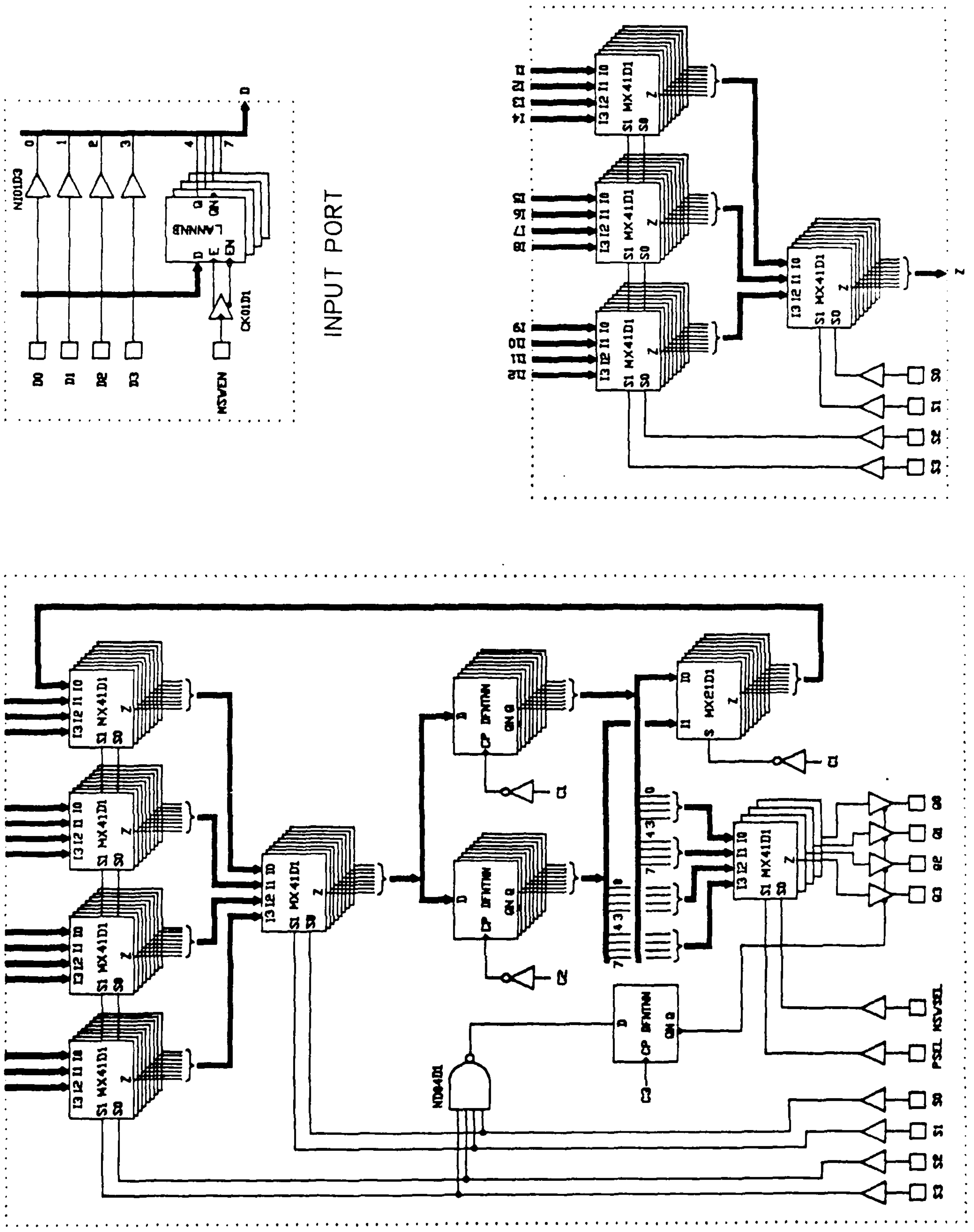
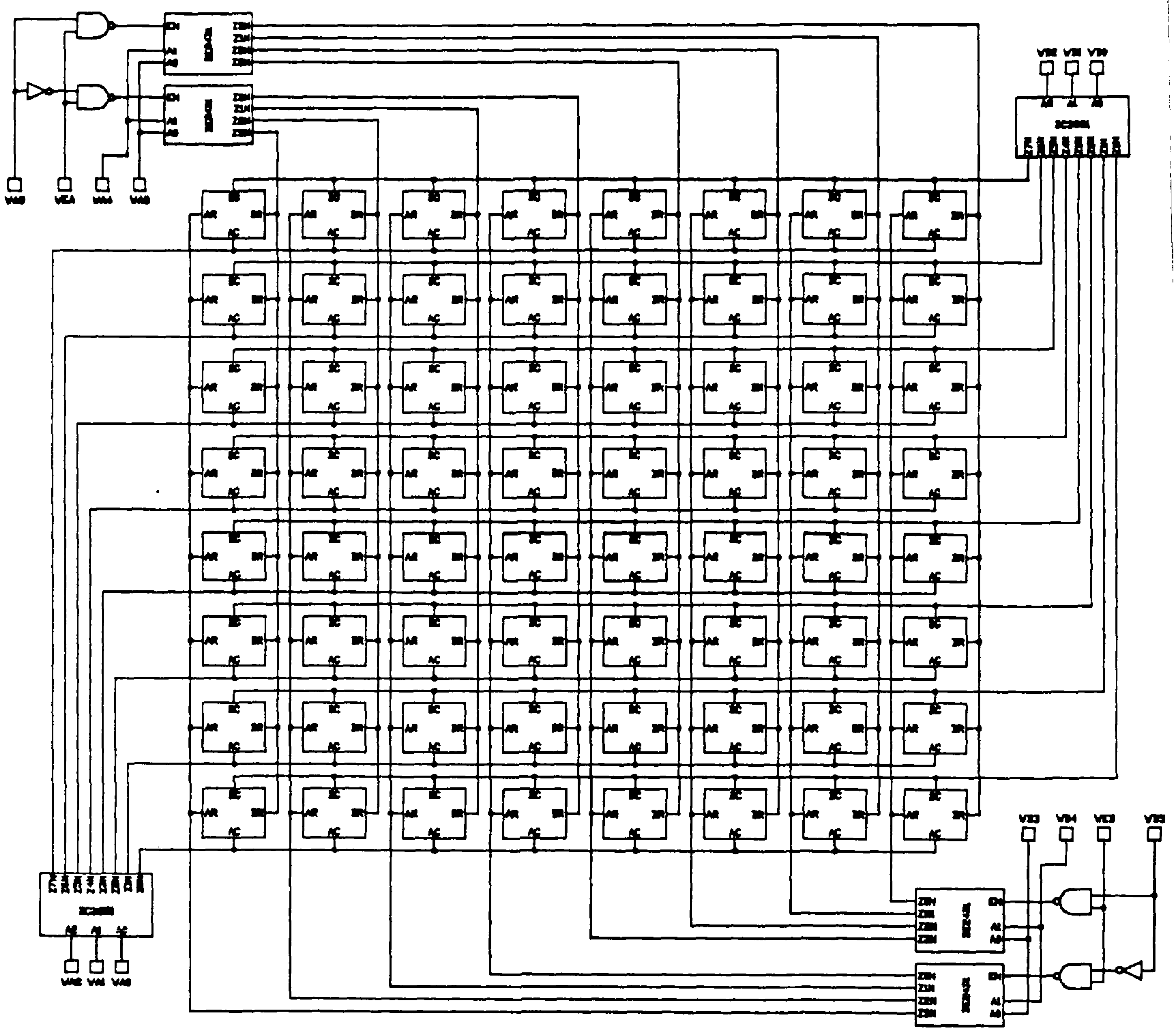
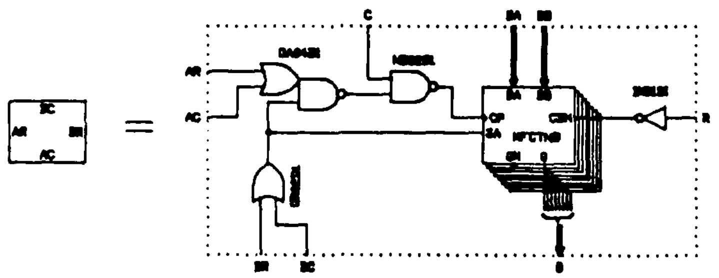


Figure 13. Crossbar Module





FOUR PORT REGISTER FILE
WRITE DECODER CIRCUITRY

Figure 14. Crossbar Module

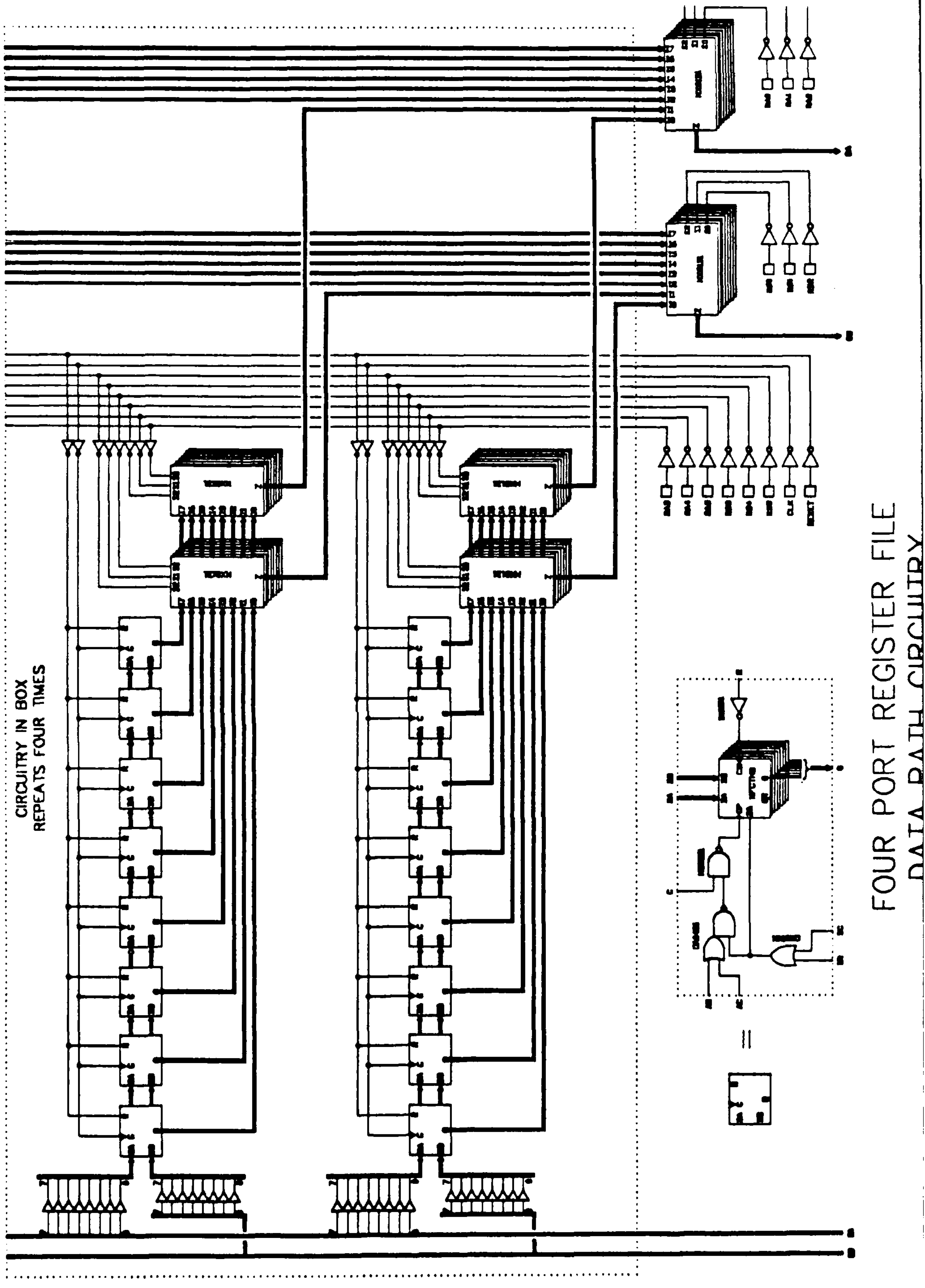


Figure 15. Crossbar Module

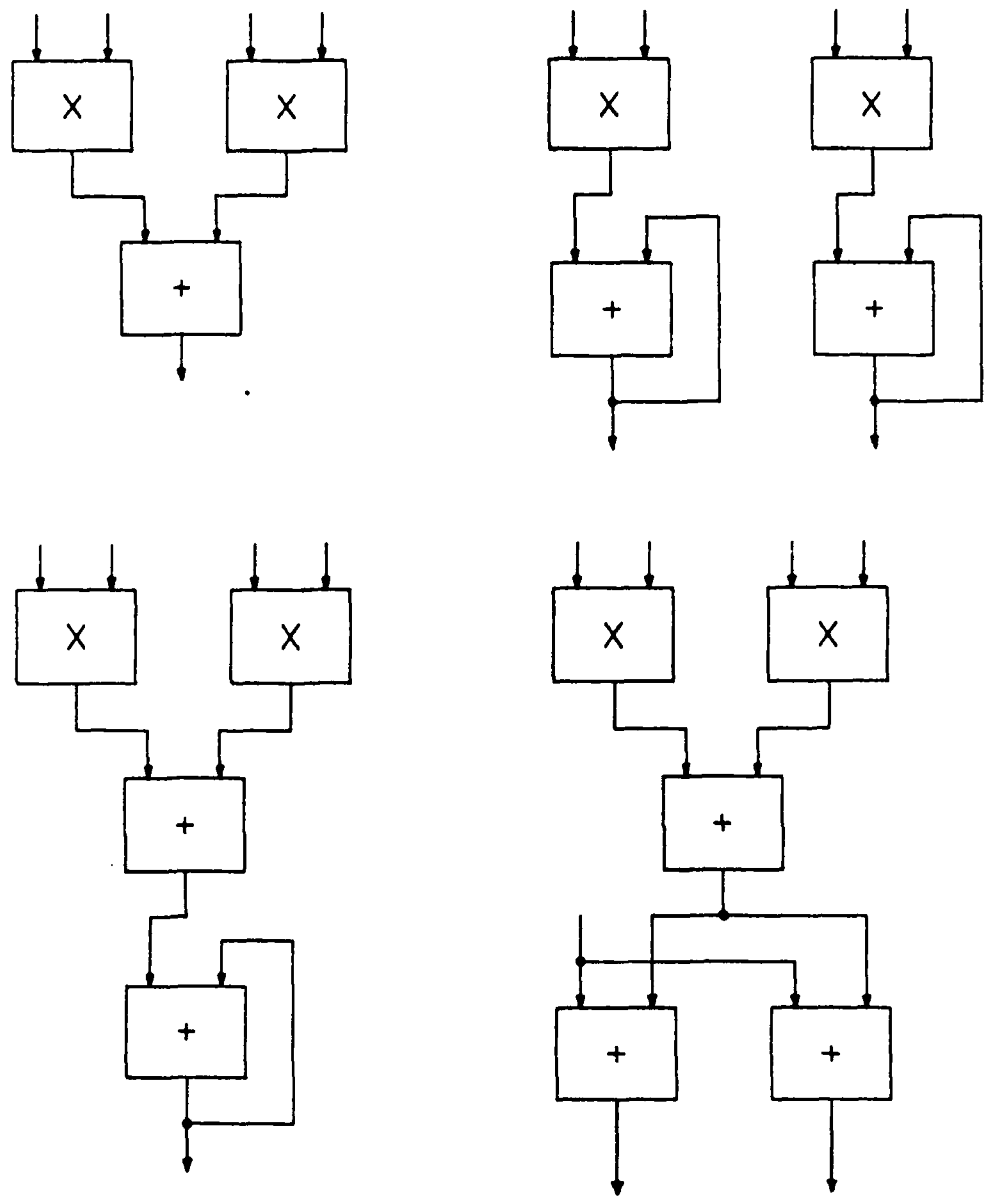


Figure 16. Dynamic ALU Configurability

5.5 STARBURST Configuration

Peak Teraop throughput can be achieved with massively paralleling the individual TPHs. Several technical challenges must however be addressed. First, synchronization must be maintained among as many TPHs as possible. Async operation degrades the useful sustained throughput due to the overhead attached to the packet-like data transfers. An interconnection strategy, called STARBURST, shown in Figure 17, overcomes the otherwise severe data skew problem.

Its starburst pattern ensures that at least 16 TPHs can be synchronized. The limiting factor is the manufacturability of the optical interconnects. In STARBURST, HYPERBARS as shown in Figure 17 are used for both connecting individual TPHs and, also, paralleling groups of 16 to a maximum of 64 levels. In second generation configurations, Space Tech proposes to uses photonic coupling among the STARBURSTS with a vertical light beam (3) split 8 ways simultaneously to 8 TPH tagged-architecture processor boards radially equidistant from the center. To maintain 10^{12} 64-bit words per sec transfer rates, a 100-bit data, command, and tag channel is required. If GaAs transceivers are used (4), fiber optic cabling is then feasible. However, 900 fiber optic wires must be threaded through the center of each STARBURST. This is pushing repeatedly reliable manufacturing limits.

A single physical STARBURST is very small with overall dimensions under one cubic foot. Using 4 BITt 3130-like ECL FPUs, 32 such devices will generate 1000 watts. By 1992, we can conservatively expect equivalent BiCMOS FPUs with a subsequent power reduction to 200 watts. Hence, complex hydroflourocarbon cooling will be replaced by normal convective cooling (a severe problem for the CRAY-3 with GaAs).

System structure is related to required communication protocols between processor elements. Several protocols ranging from purely local next-neighbor communication to totally global (broadcast) communication will be studied. The first step in describing the effectiveness of a system structure is to describe the initial structure. Communication details necessary for system operation are important for an adequate description.

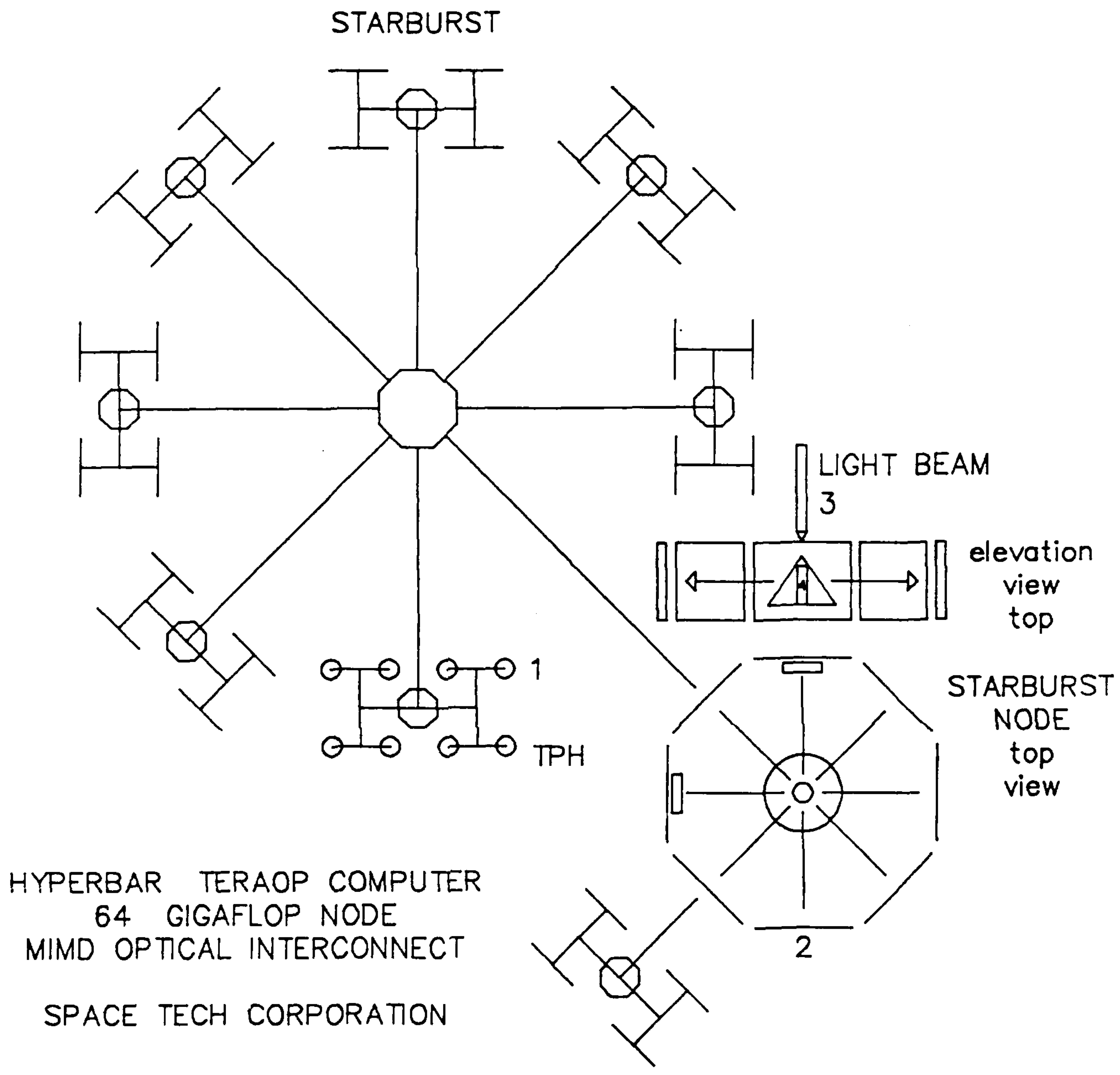


Figure 17. STARBURST Organization

5.6 HYPERBAR Interconnection

The parallel operation of TENSOR nodes in synchrony is accomplished with fiber-optic wiring. Each TENSOR FPU board is coupled in the center of each TPH board with a vertical stripline. Electronic logic (FPU's, cache, register file) is interfaced via a 64-bit parallel ASIC XBAR port to the system bus with 100 fiber-optic cables. 64 cables are assigned to the 64 data bits for parallel transfer. 36 remaining cables carry "packet" control and timing information to/from each tagged-architecture TENSOR node. Tagging enhances the global broadcast to all TPHs simultaneously. (Individual processors respond only to their tag or ID.) Photo-optical coupling is facilitated by a transparent spindle in the center of 8 TPHs. With this physical arrangement, 1012 bits/sec I/O is possible with imperceptible clock skew.

5.7 Host Interface

During development of the TENSOR single node, an interface to a PC will be used. The design of this high speed interface allows the microprogrammers to upload and download microcode to the TENSOR architecture so that rapid development of the specific SDI algorithms can be completed. The circuit is shown in Figure 18.

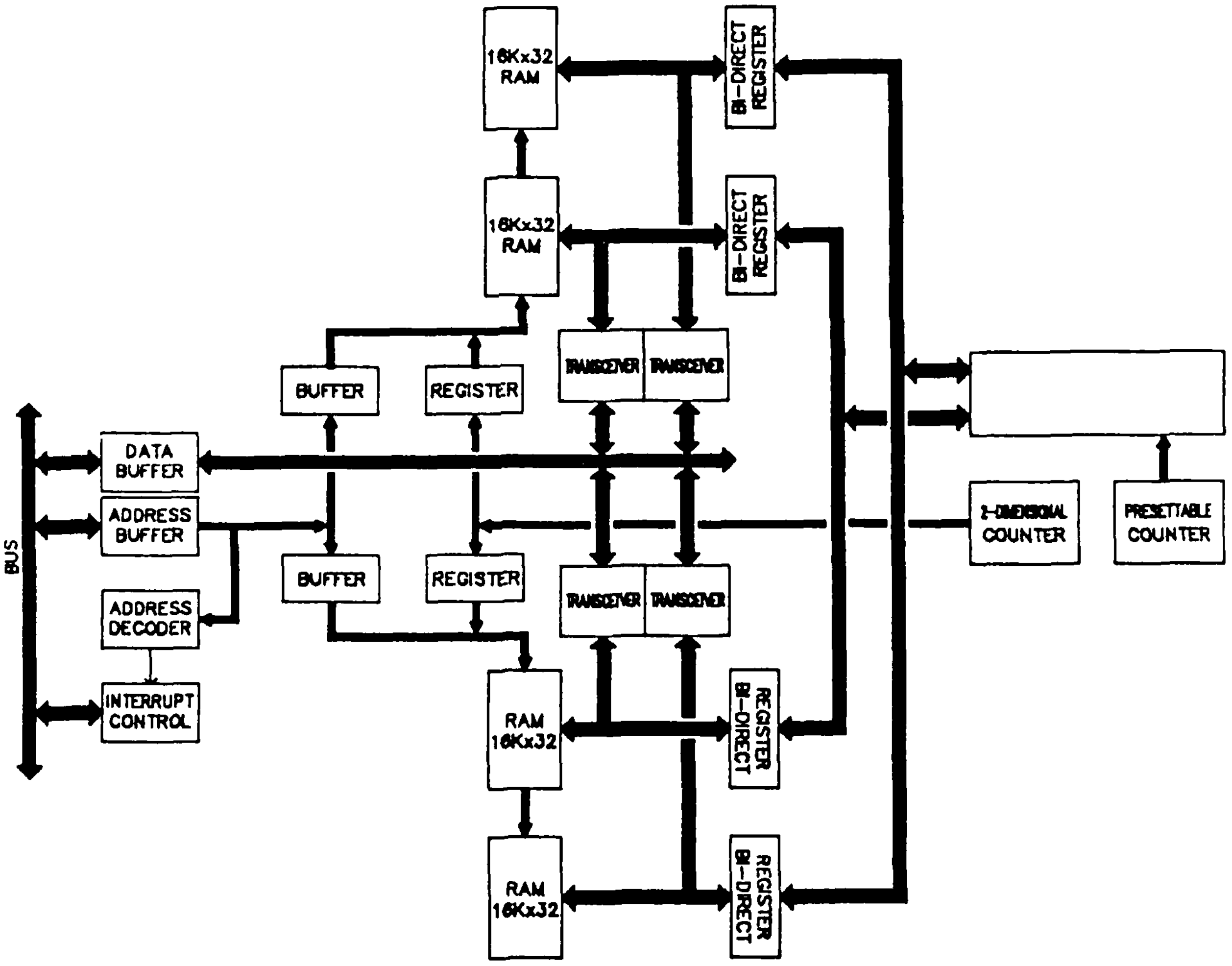


Figure 18. Host Interface

Here, TENSOR is attached to 4 bidirectional counters so that rapid transfer of microcode can occur. It is anticipated that several thousands of microinstructions will be needed for the SVD, FFT, AFT, and WV transforms.

Necessary buffers, transceivers, and decoders are used to effect the transfer across the two computer buses. The design is essentially a memory-mapped approach that can be altered to DMA transfer when needed. Four 16kx32 RAMs are used to buffer data which is arriving at dissimilar clock speeds of the two machines. Both computers will be able to process while data is being transferred, thus making overlapped I/O possible.

6.0 Recommendations and Conclusions

6.1 Algorithms

A transform that takes the data into the frequency domain whether it is an FFT, CHIRP Z, or AFT appears to co-dominate all computations of the selected algorithm suite in this effort with an ultrafast matrix inversion. Even the ISAR simplification according to Snyder shows the utility of the FFT although indirectly. (We know that ISAR can be executed by convolutions. But, if the convolutions are greater than 128 points long, then the FFT followed by an inverse FFT is actually faster computationally.) Hence, an architecture which minimally executes a frequency transformation very fast is attractive. The Fourier transform alone is not sufficient to guarantee success in Phase III because the optimal architecture must consider the massive data movement of screen sizes approaching 2k by 2k at 32 bit pixels. Optimal memory configurations are necessary. Hence, Space Tech proposes the n-port organizations.

Space Tech recommends that the Chirp-Z transform not be currently considered as a viable alternative to the FFT. If, however, more efficient computational sequences can be found for the CZT, then this issue will be reopened.

The dataflow architecture for an FFT should be examined carefully and its dataflow requirements should be integrated with those of the other algorithms. If the Austek architecture is fast enough, then Space Tech will capture the reduction of its distributive control architecture, hoping to reduce chip area requirements. Otherwise, Space Tech intends to capitalize on the CORDIC implementations of FFTs because these architectures may be less rigid to other algorithms.

6.2 Architectures

From the analyses todate, evidence is growing to suggest that the Wigner-Ville, Arithmetic Fourier Transform, and an SVD form the crucial algorithms to be embedded on a custom architecture to be designed for the execution of the several other algorithms in this project. This recommendation is based upon the simplicity of the algorithm, sparse usage of slow multiplications, regularity of the computations, and the simple dataflow of dependent data. This last reason also is attractive for the eventual implementation of an ASIC or custom gate array. There is additional evidence that the AFT can be executed rapidly in a systolic array because the data dependency paths are relatively sparse.

An N-Port memory connected to 4 ALU architecture easily outperforms several 1990 minisupers. This shared memory architecture is both an SIMD and an MIMD architecture with ALU pipes and parallel shared local memory.

REFERENCES

[Abis88] J. Abiss, "Super-Resolution for SDI Applications," presentation at the SDIO/ISTO Symposium, Washington D.C., June 20-24, 1988.

[BaIp85] J.L. Barlow, and I.C.F. Ipsen, "Scaled Givens Rotations for the Solution of Linear Least Squares Problems on Systolic Arrays (Revised Version)," Technical Report CS-85-05, Pennsylvania State University, Dept. of Computer Science, 1985.

[Bern84] M. Bernfeld, CHIRP Doppler Radar," Proc. IEEE, Vol. 72, No. 4, pp. 540-541, April 1984.

[Boas87] B. Boashash, "An Efficient Real-Time Implementation of the Wigner-Ville Distribution," IEEE Trans. on Acoustics, Speech, and Signal Processing, Vol. ASSP-35, November 1987, pp. 1611-1618.

[Boas86] B. Boashash and H.J. Whitehouse, "Seismic Applications of the Wigner-Ville Distribution," Conference on Circuits and Systems, 1986, pp. 34-37.

[Boas89] B. Boashash, "Use of the Wigner-Ville Distribution to Separate Seismic Events and to Analyse Reflected Vibroseis Signals in the Time-Frequency Plane," ICASSP, April 1989.

[Boua79] B. Bouachache, et al., "Sur la Possibilite d'Utilizer la Representation Conjointe en Temps et Frequence de Ville aux Signaux Vibroseismeques," Colloque Nat. Trait. Sig., Fretsi, Pages 121-1, 121-6, 1979.

[Boud87] G.F. Boudreaux-Bartels and T.W. Parks, "Discrete Fourier Transform Using Summation by Parts," ICASSP87, April 1987.

[Boud89] G.F. Boudreaux-Bartels, D.W. Tufts, P. Dhir, G. Sadasiv, and G. Fischer, "Analysis of Errors in the Computation of Fourier Coefficients Using the Arithmetic Fourier Transform (AFT) and Summation by Parts (SBP)," 1989 Int. Conf. on Acoustics, Speech and Signal Processing, pp. 1011-1013, May 1989.

[Bren82] R.P. Brent and F.T. Luk, "A Systolic Architecture for the Singular Value Decomposition," Report TR82-522, Dept. of Computer Science, Cornell University, 1982.

[BrLu82] R.P. Brent and F.T. Luk, "A Systolic Architecture for the Singular Value Decomposition," Report TR-CS-82-09, Dept. of Comp. Science, The Australian National University, 1982.

[BrLV83] R.P. Brent, F.T. Luk and C.P. Van Loan, "Computation of the Singular Value Decomposition using Mesh-Connected Processors," Report TR-82-528, Dept. of Comp. Science, Cornell University, 1983.

[Chang90] P.R. Chang and C.S. G. Lee, "A Decomposition Approach for Balancing Large-Scale Acyclic Data Flow Graphs," IEEE Trans. on Computers, Vol. 39, No. 1, January 1990, pp. 34-46.

[Ewer90] L. M. Ewerbring and F.T. Luk, "Computing the Singular Value Decomposition on the Connection Machine," IEEE Trans. on Computers, Vol. 39, No. 1, January 1990, pp. 152-155.

[FeHa86] K.V. Fernando and S.J. Hammarling, "Systolic Array Computation of the SVD of Complex Matrices," Proc. SPIE 696, pp. 54-61, 1986.

[Golu70] G.H. Golub and C. Reinsch, "Singular Value Decomposition and Least Squares Solutions," Numer. Math., Vol. 14, pp. 403-420, 1970.

[GoVa83] G. Golub and C. Van Loan, Matrix Computations, Johns Hopkins University Press, Baltimore, MD, Section 12.3 on "Total Least Squares," pp. 420-425, 1983.

[Haac89] E.M. Haacke, Z. Liang, and S.H. Izen, "Superresolution Reconstruction Through Object Modeling and Parameter Estimation," IEEE Trans. on ASSP, Vol. 37, No. 4, April 1989, pp. 592-595.

[Hell81] D.E. Heller and I.C.F. Ipsen, "Systolic Networks for Orthogonal Decompositions with Applications," Proc. of Conf. on Advanced Research in VLSI, 1981.

[Ips84] I.C.F. Ipsen, "Singular Value Decomposition with Systolic Arrays," Proc. SPIE 495, pp. 13-21, 1984.

[JeLW] J.F. Jensen, L.E. Larson, and M. Waldner, "High Speed Digital Computation with GaAs Technology," private manuscript.

[Luk80] F.T. Luk, "Computing the Singular Value Decomposition on the Illiac IV," ACM Trans. on Mathematical Software, Vol. 6, pp. 524-539, 1980.

[LukF82] F.T. Luk, A.M. Finn, and C. Pottle, "Systolic Array Computation on the Singular Value Decomposition," SPIE, Vol. 341, Real Time Signal Processing V, paper 341-05, 1982.

[Lund87] S.F. Lundstrom, "Applications Considerations in the System Design of Highly Concurrent Multiprocessors," IEEE Trans. on Computers, Vol. C-36, No. 11, November 1987, pp. 1292-1309.

[Midw87] J.E. Midwinter, "High-speed Synchronous Wideband Switching Matrices," IEE Proceedings, Vol. 134, Pt. J, No. 5, October 1987, pp. 261-268.

[Mokh89] N. Mokhoff, "Bi FET GaAS Suitable For Integration," EE Times Tech File, October 1989.

[MoLa86] J.H. Moreno and T. Lang, "A Multilevel Pipelined Processor for the Singular Value Decomposition," Proc. SPIE 698, pp. 100-112, 1986.

[NaPH86] J.G. Nash, K.W. Przytula and S. Hansen, "Systolic/Cellular Processor for Linear Algebraic

Operations," Proc. IEEE ASSP Workshop on VLSI Signal Processing, Los Angeles, pp. 306-315, 1986.

[Nash79] J.G. Nash, Compact Numerical Methods for Computers, Halsted Press, John Wiley & Sons, New York, 1979.

[Paig81] C.C. Paige and M.A. Saunders, "Towards a Generalized Singular Value Decomposition," SIAM Journal on Numerical Analysis, Vol. 18, pp. 398-405, 1981.

[Schr82] R. Schreiber, "Systolic Arrays for Eigenvalue Computation," SPIE, Vol. 341, Real Time Signal Processing, paper 341-50, 1982.

[Schw73] H.R. Schwarz, H. Rutishauser, and E. Steifel, Numerical Analysis of Symmetric Matrices, Prentice-Hall, 1973.

[Snyd86] D.L. Snyder, H. Whitehouse, T. Wohlschlaeger, and R. C. Lewis, "A New Approach to Radar/Sonar Imaging," SPIE Vol. 696 Advanced Algorithms and Architectures for Signal Processing, pp. 134-136, 1986.

[Spie83] J.M. Speiser and H.J. Whitehouse, "A Review of Signal Processing With Systolic Arrays," SPIE, Vol. 431, Real Time Signal Processing VI, pp. 1-6, 1983.

[TrGi79] J.G. Tront and D.D. Givone, "Multiple-Valued Logic Gates Using MESFETS," Proc. 9th Symp. Multiple-Valued Logic, pp. 175-181, 1979.

[Upa80] L.C. Upadahyayula, "GaAs MESFET Comparators for Giga Bit rate Analog to Digital Converters," RCA Rev., Vol. 41, June 1980.

[Vant71] H. L. Van Trees, "Detection, Estimation, and Modulation Theory, Part III," John Wiley and Sons, New York, 1971.

[Whi87] S.A. White, "An Architecture for a 16-Point FFT GaAs Building Block," Proc. 21st Asilomar Conf. on Signals, Systems and Computers, 1987.

[Wilk65] J.H. Wilkinson, The Algebraic Eigenvalue Problem, Oxford University Press, 1965.

[Wolf69] E. Wolf, "Three Dimensional Structure Determination of Semitransparent Objects from Holographic Data," Opt. Commun., Vol. 1, pp. 153-156, 1969.

[WuTo87] R. Wu and M.N. Toksoz, "Diffraction Tomography and Multisource Holography Applied to Seismic Imaging," Geophysics, Vol. 52, No. 1, January 1987, pp. 11-25.

[Zol87] M.D. Zoltowski, "Signal Processing Applications of the Method of Total Least Squares," Proc. 21st Asilomar Conf. on Signals, Systems and Computers, 1987.